1 + N Network Protection for Mesh Networks: Network Coding-Based Protection Using p-Cycles

Ahmed E. Kamal, *Senior Member, IEEE*

***Abstract*—p-Cycles have been proposed for preprovisioned $1:N$ 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 to the destination on two link disjoint circuits, hence recovery from failures is expeditious. However, this requires a large number of protection circuits. In this paper, we introduce a new concept in protection: $1+N$ protection, in which a p-Cycle, similar to FIPP p-cycles, can be used to protect a number of bidirectional connections, which are mutually link disjoint, and also link disjoint from all links of the p-Cycle. However, data units from different circuits are combined using network coding, which can be implemented in a number of technologies, such as Next Generation SONET (NGS), MPLS/GMPLS, or IP-over-WDM. The maximum outage time under this protection scheme can be limited to no more than the p-Cycle propagation delay. It is also shown how to implement a hybrid $1+N$ and $1:N$ protection scheme, in which on-cycle links are protected using $1:N$ protection, while straddling links, or paths, are protected using $1+N$ protection. Extensions of this technique to protect multipoint connections are also introduced. A performance study based on optimal formulations of the $1+1$, $1+N$ and the hybrid scheme is introduced. Although $1+N$ speed of recovery is comparable to that of $1+1$ protection, numerical results for small networks indicate that $1+N$ is about 30% more efficient than $1+1$ protection, in terms of the amount of protection resources, especially as the network graph density increases.**

***Index Terms*—1 + N protection, network coding, optical networks, p-Cycles, protection, survivability.**

I. Introduction

With the use of optical fibers in network backbones, large amounts of bandwidth are provided on a single fiber, and huge amounts of traffic are carried on the fiber, The failure of a single fiber, which is not uncommon, 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*.

Research on techniques to provide optical network survivability has received special attention. Techniques for optical network survivability can be classified as *Predesigned Protection* and *Dynamic Restoration* techniques [1]. In predesigned protection, which is a proactive technique, bandwidth is reserved in advance so that when a failure takes place, backup paths which are preprovisioned, are used to reroute the traffic affected by the failure. These techniques include the $1+1$ protection, in which traffic of a lightpath 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 a 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. Protection techniques are widely used in SONET ring architectures [1]. Under dynamic restoration, which is a reactive strategy, 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.

Recently, the concept of p-Cycles has been introduced in [2]–[4], to emulate the protection techniques of SONET ring networks, and they 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.

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 computations 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. This strategy can be implemented at a number of layers.

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 in fact 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:

Manuscript received August 13, 2007; revised September 07, 2008 and March 15, 2009; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor J. Yates. First published August 11, 2009; current version published February 18, 2010. This work was supported in part by the National Science Foundation under Grants CNS-0626741 and CNS-0721453 and by a gift from Cisco Systems. Parts of this paper were presented at IEEE Globecom 2006 and IEEE Globecom 2007.

The author is with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011 USA (e-mail: kamal@iastate.edu).

Digital Object Identifier 10.1109/TNET.2009.2020503

1) Protection against single link failure is guaranteed.
2) p-Cycles which are typically employed for $1:\mathrm{N}$ protection, are used to provide $1+\mathrm{N}$ 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 shared.
3) Resuming data reception on the protection path is guaranteed to be within 1.5 times the propagation delay around a p-Cycle, but can be much less than this limit.

In addition, and as a byproduct, in the absence of failures, this scheme provides an error recovery functionality in the absence of failures. This will be discussed in Section V.

In this paper we introduce the basic concepts and theoretical bases of the strategy, and how it can be used to provide $1+\mathrm{N}$ protection using p-Cycles against single link failures. We discuss the implementation of this scheme in a number of technologies and layers in Section VI.

The rest of the paper is organized as follows. In Section II we provide a brief background on 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 for encoding and decoding data units on p-Cycles in order to provide protection for bidirectional unicast connections using one bidirectional p-Cycle. We illustrate this procedure using an example. We also outline the advantages of this scheme, as well as other uses for this scheme, especially in error control. In Section VI 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 1.5 times the delay on the p-Cycle. Some other implementation considerations, as well as notes on implementing this strategy in different technologies and protocols will also be discussed. A hybrid $1+\mathrm{N}$ and $1:\mathrm{N}$ protection scheme is introduced in Section VII in order to enable the p-Cycle to protect transmissions carried on the links used by the cycle itself. Section VIII shows some extensions to the proposed strategy which enables it to work with multipoint sessions. In Section IX we introduce an empirical comparison between $1+1$ and $1+\mathrm{N}$ protection. We also introduce a comparison between $1+1$ and the hybrid scheme. The comparison is based on the cost of the network in terms of the number of links, and optimal formulations for these problems are given in the Appendices. Finally, in Section X we conclude the paper.

II. Background

A. Background on p-Cycles

The p-Cycle concept [2]–[4] 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 Fig. 1). Since the protection capacity can be used to protect multiple connections, the p-Cycle belongs to the

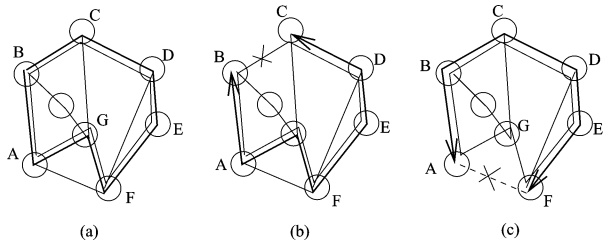

Fig. 1. p-Cycle concept. (a) A cycle (thick lines) traversing nodes A-G, and protecting circuits (thin lines) on the same physical path as the cycle, and on straddling paths. (b) Prtection of a failure on the cycle. (c) Protection of a failure on a straddling path.

$1:\mathrm{N}$ 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 Fig. 1, and *node-encircling p-Cycles*, which protect paths traversing a certain node against the failure of such a 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 amount of spare capacity, e.g., see [4, Ch. 10]. 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 preprovisioning 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 [5].

Recently, p-Cycles have been extended from protecting spans or segments of flows, to protect entire flows, i.e., end-to-end connections, regardless of the actual location of failure on the connection's working path, hence, the name Failure-Independent Path-Protecting (FIPP) p-Cycles [6], [7]. This requires all connections to be mutually link disjoint. In this case, if a connection is totally straddling or totally on the p-Cycle, a failure on the connection can be recovered from by switching the two end nodes of the connection to use the part of the p-Cycle that is disjoint from the connection (the entire p-Cycle in the case of a totally straddling connection, hence protecting twice as much working capacity on the straddling connections). However, if the connection is partly on the cycle and partly straddling, a failure is usually recovered from by using one default segment of the p-Cycle, unless the failure is on this segment; in the the latter case the complementary segment of the p-Cycle is used. This strategy leads to failure-independent end-to-end connection protection using a set of fully preconfigured protection circuits.

In this paper, we will use p-Cycles to protect a number of link disjoint connections, similar to FIPP p-Cycles, against failures. However, the protection will be done in $1+\mathrm{N}$ manner, rather an $1:\mathrm{N}$. That is, our approach is to allow two transmissions of the same signal. One transmission is on the working path, and the second one is on a protection circuit, implemented by a p-Cycle. Multiple link disjoint connections transmit their signals simultaneously on the p-Cycle. The receivers receive these two copies, and select the better of the two signals. The backup signals are

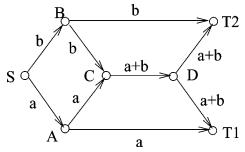

Fig. 2. An example of network coding.

transmitted simultaneously and on the same protection circuit using the technique of network coding. Our approach can also be used at a number of layers including the SONET layer, especially Next Generation SONET, ATM, MPLS/GMPLS, and IP.

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 service mode is multicast. An example of the use of network coding is shown in Fig. 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 $\mathrm{a}+\mathrm{b}$. 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.* [8]. The problem of network coding was formulated as a network flow problem in [9] and a link cost function was included in the formulation in [10]. Reference [11] 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 [12] introduced deterministic and randomized algorithms for the construction of network codes, which had polynomial time complexity. The algorithms could be used for multiple multicast sessions, where intermediate nodes may decode, and re-encode the received information. Reference [13] 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.

- In this paper, we deal with connections. A connection may consist of a circuit on a single link, or may consist of a sequential set of circuits on multiple links, e.g., a lightpath. Therefore, link protection is a special case of this technique.
- The term link is used to refer to a fiber connecting two nodes. Each link contains a number of circuits, e.g., wavelength channels, or even channels with smaller granularities, e.g., DS3.
- A p-Cycle protecting a number of connections passes through all end nodes of such connections, similar to FIPP p-Cycles. In doing so, the p-Cycle 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 all end nodes of the connections.
- We assume that all connections are bidirectional, and connections that are protected by the same p-Cycle are mutually link disjoint.
- It is assumed that data units are fixed in size.[1]
- The scheme presented in this paper is designed to protect against a single link failure. That is, when a link fails, it will be protected, and will be repaired before another link fails.
- When a link carrying active circuits fails, the tail node of the link is capable of identifying the failure in some way, e.g., by receiving empty data units.

This paper presents the concepts of using network coding on p-Cycles to achieve $1 + \mathrm{N}$ protection. It is to be noted that this strategy can be implemented using a number of layers and protocols, including the GFP [14] protocols of NGS, where data units to be treated like packets by GFP. The strategy can also be implemented using ATM, MPLS or IP.

It should be pointed out that all addition operations $(+)$ in this paper are over $\mathbb{GF}(2)$, i.e., as **modulo two additions**, i.e., Exclusive-OR (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 connections.

The example is shown in Fig. 3(a), and there are three unidirectional connections from source D_i to destination U_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 to include all the 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, D_i, and the other by the receiver, U_i. To distinguish between those last two data units we refer to

[1]The case of variable size data units will be discussed in Section VI.

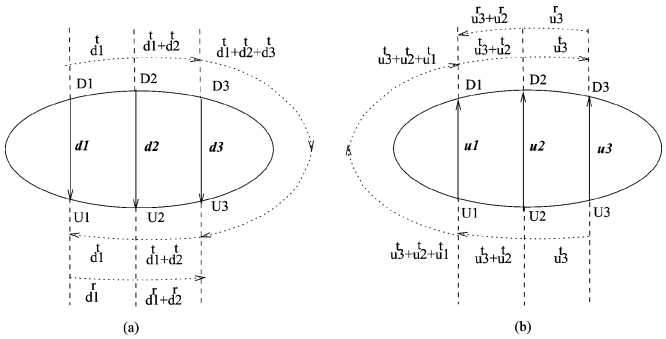

Fig. 3. An example of the use of network coding on p-Cycles to protect against single link failures. (a) The sources are at D_i, and the destinations are at D_i nodes. (b) The source are at D_i, and the destinations are at D_i nodes.

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 D_1 transmits d_1^t in the clockwise direction. Node D_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 D_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 U_3 receives $d_1^t + d_2^t + d_3^t$ on the p-Cycle, and in the clockwise direction.
2) On the same direction of the p-Cycle, but at the destinations, when destination U_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 U_2. Node U_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 U_1. U_1 removes d_1^t from the clockwise cycle.
3) Also, when node U_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 U_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, U_i, for $i = 1, 2, 3$, receives two copies of d_i:

1) One copy on the primary 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 Section VI.

When a failure occurs, it will affect at most one working path, e.g., working path i. In this case, we assume that U_i will receive an empty data unit on the working path. Therefore, U_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.

The case in which information is sent in the opposite direction, i.e., from D_i to D_i is shown in Fig. 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 D_i and U_i that we enforced in this example. In the general case where D_i and U_i can be arbitrarily ordered, as will be shown next, combining the two bidirectional sessions would require two full p-Cycles. However, by linearly combining u_i and d_j signals on the same link and in the same direction, it is possible to reduce the number of p-Cycles to one full cycle, hence the name $1 + \mathrm{N}$ protection, where one full p-cycle is used for protection N connections. This will be illustrated in the next section.

V. Network Coding Strategy on p-Cycles

In this section, we introduce our general strategy for achieving $1 + \mathrm{N}$ protection in mesh networks using p-Cycles.

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 $\mathbb{A} = \{A_i \,|\, 1 \le i \le N\}$ and $\mathbb{B} = \{B_i \,|\, 1 \le i \le N\}$.[2] We denote the data units transmitted from nodes in $\mathbb{A}$ to nodes in $\mathbb{B}$ as d units, and the data units transmitted from nodes in $\mathbb{B}$ to nodes in $\mathbb{A}$ as u units.

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

$$\sum_{j, A_j \in \mathbb{A}'} u_j' \tag{1}$$

$$u_i' + \sum_{j, A_j \in \mathbb{A}'} u_j' \tag{2}$$

for some $\mathbb{A}' \subset \mathbb{A}$, $A_i \notin \mathbb{A}'$, where data unit u_j' is the one to be received by A_j, and the sum is modulo 2. In this case, A_i can recover u_i' by adding (1) and (2) using modulo 2 addition also.

Our procedure goes through the following steps.

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).[3] These two p-Cycles do not have to traverse the same links, but must traverse the nodes in the same order.

[2]Note that the choice of the labels A_i and B_i is arbitrary, as long as A_i and B_i communicate with each other.

[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, and then apply the strategy to those connections.

Algorithm 1: Algorithm for constructing the sequences $\mathcal{D}$ and $\mathcal{U}$

Initialization:

$\mathcal{D} = \mathcal{U} = (\ )$; //initialize empty sequences
$i = 1,\ j = N$;
$\mathbb{C} = \mathbb{A} \cup \mathbb{B}$;
$D_1 = A_1$;
// select first node in $\mathcal{D}$, 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 $\mathcal{D}$* **then**
    $U_j = c$;
    $j = j - 1$;
  else
    $D_i = c$;
    $i = i + 1$;
  $\mathbb{C} = \mathbb{C} - \{c\}$;

2) Construct two sequences of nodes, $\mathcal{D} = (D_1, D_2, \ldots, D_N)$ and $\mathcal{U} = (U_1, U_2, \ldots, U_N)$ of equal lengths, N. All elements of $\mathcal{D}$ and $\mathcal{U}$ 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 $\mathcal{D}$ to be in the clockwise direction, and the sequence of nodes in $\mathcal{U}$ to be in the counter-clockwise direction. We also start with any node[4] in $\mathbb{C}$ as D_1, and we label this node as A_1. All nodes in $\mathcal{D}$ belong to the set $\mathbb{A}$, and all nodes in $\mathcal{U}$ belong to the set $\mathbb{B}$. Node U_1 will always be the one to the left of node D_1. The example in Fig. 4 shows how ten nodes, in five connections are assigned to $\mathcal{D}$ and $\mathcal{U}$.
   A node D_i in $\mathcal{D}$ (U_i in $\mathcal{U}$) transmits d_i (u_i) data units to a node in $\mathcal{U}$ ($\mathcal{D}$).
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, $\mathbf{T}$. On this half cycle newly generated d_i units generated by nodes in $\mathcal{D}$, and newly generated u_i units generated by nodes in $\mathcal{U}$ 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 $\mathcal{U}$ and $\mathcal{D}$, respectively.
   b) A half p-Cycle in the counter-clockwise direction, $\mathbf{R}$. On this half cycle, d_i units received on the primary working paths by nodes in $\mathcal{U}$, and u_i data units received, also on the primary working paths, by nodes in $\mathcal{D}$ 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 $\mathcal{D}$ and $\mathcal{U}$, respectively.

[4]The selection of the node to be labeled D_1 is important in bounding the delay to recover from lost data due to failures, and also the outage time. This issue will be discussed in Section VI.

   Note that the encoding and decoding operations referred to above are simple modulo 2 addition operations of data units to be transmitted and the data units 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. u_i^t data units encoded together mus also belong to the same round. Rounds on the $\mathbf{T}$ cycle can be started by the D_1 node. Other nodes follow D_1 and transmit their own d_i and u_i data units which belong to the same round. Rounds in the $\mathbf{R}$ cycle are also started by node D_1, but node U_1 is the first node to transmit in a round, followed by other nodes in the counter-clockwise direction. All nodes in $\mathcal{D}$ and $\mathcal{U}$ 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. The handling of round numbers, and which data units to transmit in round n, will be explained in detail in Section VI.E.

2) Encoding Operations: The network encoding operation is executed by the nodes in $\mathcal{D}$ and $\mathcal{U}$ as follows (assuming no link failures):

1) Node D_i:
   a) The node will add the following data units to the signal received on $\mathbf{T}$:
      - Data unit d_i^t, which is newly generated by D_i.
      - Data unit u_j^t, which is received on the primary path from U_j.

      The result is transmitted on the outgoing link in $\mathbf{T}$.
   b) The node will add the following data units to the signal received on $\mathbf{R}$, and will transmit the result on the outgoing link in $\mathbf{R}$.
      - 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 U_j.
2) Node U_i will perform similar operations:
   a) The node will add the following data units to the signal received on $\mathbf{T}$:
      - Data unit u_i^t, which is newly generated by U_i, and
      - Data unit d_j^t, which is received on the primary path from D_j.

      The result is transmitted on the outgoing link in $\mathbf{T}$.
   b) The node will add the following data units to the signal received on $\mathbf{R}$:
      - 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 U_j.

      Also, the result is transmitted on the outgoing link in $\mathbf{R}$.

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

- $T(D_i)$: node in $\mathcal{U}$ transmitting and receiving from D_i.
- $S(U_i)$: node in $\mathcal{D}$ transmitting and receiving from U_i.
- $\hat{D}(Tx)_i^n$ = sum of d data units transmitted by $D_1, D_2, \ldots, D_i$ in round n and by $D_{i+1}, D_{i+2}, \ldots, D_N$ in round $n - a$ on half cycle $\mathbf{T}$ which have not yet been removed by their corresponding receiver nodes in $\mathcal{U}$. a is

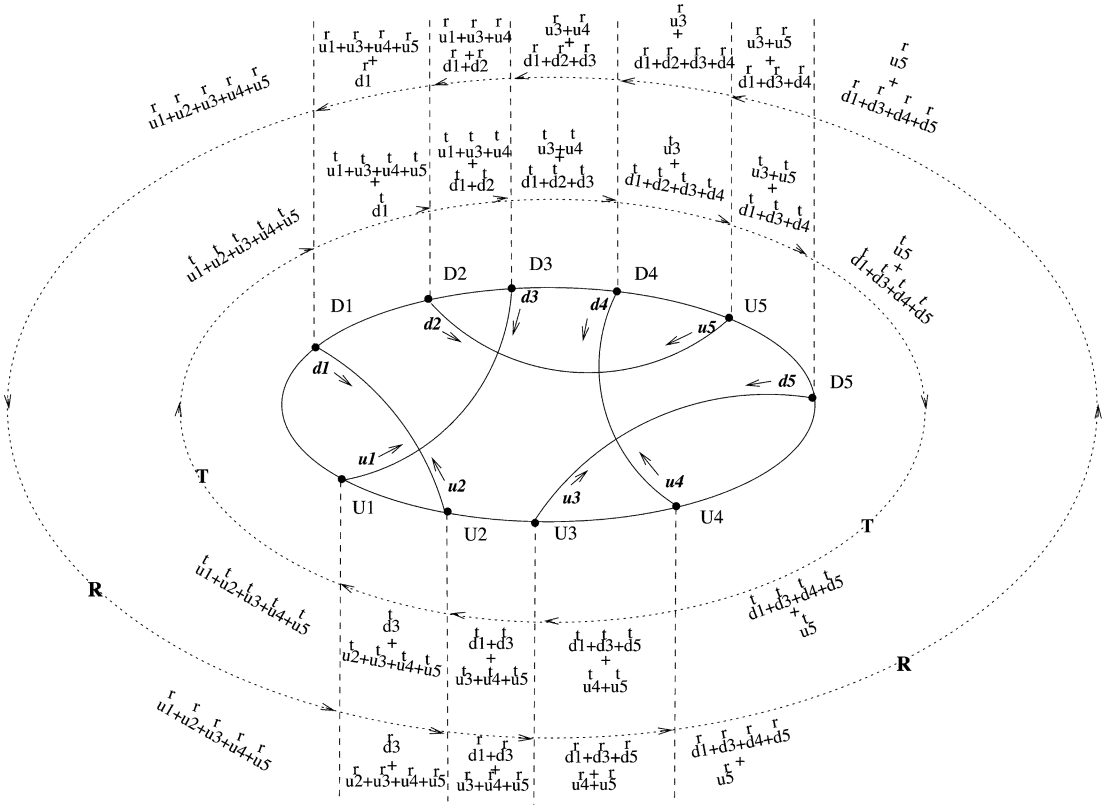

Fig. 4. An example of the application of the network coding procedure to a p-Cycle.

defined in (9), and is the cycle propagation delay in terms of packets.

- $\hat{U}(Tx)_i^n$ = sum of u data units transmitted by $U_i, U_{i+1}, \ldots, U_N$ in round n and by $U_1, U_2, \ldots, U_{i-1}$ in round $n-1$ on half cycle $\mathbf{T}$ which have not yet been removed by their corresponding receiver nodes in $\mathcal{D}$.
- $\hat{U}(Rx)_i^n$ = sum of u data units received by $D_i, D_{i+1}, \ldots, D_N$ in round n and by nodes $D_1, D_2, \ldots, D_{i-1}$ in round $n-1$ and transmitted on half cycle $\mathbf{R}$ which have not yet been removed by their corresponding receiver nodes in $\mathcal{U}$.
- $\hat{D}(Rx)_i^n$ = sum of d data units received by $U_1, U_2, \ldots, U_i$ in round n and by nodes $U_{i+1}, U_{i+2}, \ldots, U_N$ in round $n-1$ and transmitted on half cycle $\mathbf{R}$ which have not yet been removed by their corresponding receiver nodes in $\mathcal{D}$.

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

1) In step 1a above, node D_i receives $\hat{D}(Tx)_{i-1}^n + \hat{U}(Tx)_j^n$ on the incoming link on $\mathbf{T}$. Node U_j is the node next to D_i in the counter-clockwise direction. For example, for D_2 in Fig. 4, it is U_1, and for D_5, it is U_5. The addition operations will add d_i to $\hat{D}(Tx)_{i-1}^n$, and will remove $u_{T(D_i)}$ from $\hat{U}(Tx)_j^n$. This will result in $\hat{D}(Tx)_i^n + \hat{U}(Tx)_j^n$ at the output of node D_i, which will be transmitted on the outgoing link on $\mathbf{T}$.
   Node D_3 in Fig. 4 adds d_3, which is transmitted on the outgoing link. However, adding u_1, where $T(D_3) = U_1$, removes it and is therefore not transmitted on $\mathbf{T}$.
2) Also, in step 1b, node D_i receives $\hat{U}(Rx)_{i+1}^n + \hat{D}(Rx)_j^n$ on the incoming link on $\mathbf{R}$. Node U_j is the node in $\mathcal{U}$ which is next to D_i in the clockwise direction. For example, in Fig. 4, for D_3 it is U_5, and for D_5, it is U_4. After the addition operation, $u_{T(D_i)}$ is added, and d_i is removed. The node outputs $\hat{U}(Rx)_i^n + \hat{D}(Rx)_j^n$ on $\mathbf{R}$.
   In Fig. 4, at node D_3, the addition of d_3 to the incoming signal on $\mathbf{R}$ removes it, while the addition of u_1, where $U_1 = T(D_3)$ adds it to the signal which is transmitted on the outgoing link on $\mathbf{R}$.
3) In step 2a, node U_i receives $\hat{U}(Tx)_{i+1}^n + \hat{D}(Tx)_j^n$ on the incoming link of $\mathbf{T}$, where node D_j is the node in $\mathcal{D}$ next to U_i in the counter-clockwise direction. For example, in Fig. 4, for U_3 it is node D_5. The addition operation adds u_i, and removes d_j, where $D_j = S(U_i)$, and produces $\hat{U}(Tx)_i^n + \hat{D}(Tx)_j^n$, which is transmitted on the outgoing link of $\mathbf{T}$.
   In Fig. 4, U_2 adds u_2, and removes d_1
4) Finally, in step 2b, node U_i receives $\hat{D}(Rx)_{i-1}^n + \hat{U}(Rx)_j^n$ on the incoming link of $\mathbf{R}$, where D_j is the node next to U_i in the clockwise direction. For example, for U_5, it is D_5, and for U_3, it is D_1.
   The addition operation adds d_j, and removes u_i, where $D_j = S(U_i)$. The result is $\hat{D}(Rx)_i^n + \hat{U}(Rx)_j^n$, which is transmitted on the outgoing link of $\mathbf{R}$.
   In Fig. 4, U_3 adds d_5, and removes u_3.

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 D_i and U_j

fails. In this case, D_i does not receive u_j on the primary path. However, it can recover u_j by adding

- $\hat{D}(Tx)^n_{i-1} + \hat{U}(Tx)^n_j$ which is received on $\mathbf{T}$,
- $\hat{U}(Rx)^n_{i+1} + \hat{D}(Rx)^n_j$, that it receives on $\mathbf{R}$, and
- d_i that it generated and transmitted earlier.

For example, at node D_3 in Fig. 4, adding the signal received on $\mathbf{T}$ to the signal received on $\mathbf{R}$, and d_3, then u_1 can be recovered, since $U_1 = T(D_3)$ generated u_1.

Similarly, node U_j can recover d_i by adding

- $\hat{U}(Tx)^n_{i+1} + \hat{D}(Tx)^n_j$ which it receives on $\mathbf{T}$,
- $\hat{D}(Rx)^n_{i-1} + \hat{U}(Rx)^n_j$ which is received on $\mathbf{R}$, and
- u_i that it generated and transmitted earlier.

Node U_2 adds the signals on $\mathbf{T}$ and $\mathbf{R}$, and the u_2 it generated earlier to recover d_1. Note that the signals on $\mathbf{T}$ and $\mathbf{R}$ which are added together must have the same round number.

B. Advantages of the Proposed Strategy

The proposed strategy has a number of advantages, which can be summarized as follows:

- The strategy provides $1 + \mathrm{N}$ protection against single link failures, in which the protection resources are shared between connections, hence resulting in a potential reduction of the protection circuits over $1+1$ protection. This is especially evident in cases where the nodal degree is high, e.g., four, such as in the NJ-LATA and Pan-European COST239 networks.
- Similar to FIPP p-Cycles, the management plane will be simplified since it does not have to detect the location of the failure.
- The control plane functionality will be simplified since it does not need to reroute the signals at any of the switches, including those at the end nodes of the failed connection, in order to 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 in the absence of link failures. For example, if the two copies do not match, then this is an indication of an error. If the copy received on the working path is corrupted (which can be detected through the frame check sequence), then the copy recovered from the p-Cycle can be used instead.
- Since data units are added together on the p-Cycle, data units encrypt each other, which provides a measure of security on the shared protection circuits at no additional cost. This requires that the number of connections protected by a p-Cycle be greater than 2.

VI. Implementation Considerations

In this section we consider issues that need to be taken into account for implementing the above strategy. These include timing considerations, detection and removal of protection channel errors, security issues, and protocol implementation.

A. 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 D_1 can start the first round on $\mathbf{T}$, and the remaining nodes $\mathcal{D}$ and $\mathcal{U}$ follow. When data in the first round arrives at node U_1 on the working circuits, it starts transmitting data received in round 1 on $\mathbf{R}$, and all the nodes in $\mathcal{U}$ and $\mathcal{D}$ follow. 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 arrive. Moreover, the primary path will have a delay which does not exceed τ, where τ is the propagation delay around the p-Cycle. Otherwise, the primary path will choose the shorter path over the cycle.

There is a number of timing and delay issues that need to be considered:

1) *Failure-Free Operation:*
   Under the above assumption of the primary path being shorter than any secondary backup path, nodes in $\mathcal{D}$ and $\mathcal{U}$ will respectively receive their u_i and d_i data units on the primary paths before they receive them on the backup paths. In this case, data units can be added to, and removed from the corresponding half p-Cycles without delay.[5]
2) *Operation Under Working Path Failure:*
   Assume that the working path between nodes D_i and U_k has failed. All other nodes will not be affected by this failure. Let us first consider the case of receiving d_i data units by U_k. Nodes in $\mathcal{D}$ can transmit their d_i data units on $\mathbf{T}$ in the corresponding cycles, and d_i data units must be removed by their corresponding receivers in $\mathcal{U}$. This can be done by all nodes similar to case 1 above.

However, for node U_k, d_i data units in cycle n received on $\mathbf{T}$ may have to be delayed at U_k until the corresponding combination of data units in cycle n on $\mathbf{R}$ arrive at U_k. To derive an upper bound on this delay, we now introduce a condition on the selection of nodes D_1 and U_1:

> Find the two end nodes of a connection, such that on one sector of the p-Cycle, there is no connection that has its two end nodes on this sector. The end node of this connection, which is at the end of this sector in the counter-clockwise direction is taken as U_1, and the next node in the clockwise direction is taken as D_1.

For example, in Fig. 4, end nodes D_3 and U_1 of a connection have the sector that includes nodes D_1 and D_2 satisfy this condition. Therefore, the end node of this connection in the counter-clockwise direction is taken as U_1. Notice also that nodes D_2 and U_5 satisfy this condition, and node D_2 could have been taken as U_1, while node D_3 would have been labeled D_1 in this case.

Now we evaluate an upper bound on the delay time at node U_k, Δ_{U_k}, which is the time that node U_k will have to delay data units on the $\mathbf{T}$ cycle. To illustrate the derivation, we will use the space-time diagram in Fig. 5, which corresponds to the example in Fig. 4. In this figure, the p-Cycle is broken between nodes D_1 and U_1, and the cycle is unrolled. It is also assumed that the

[5]In case the working path is longer than the backup path on the p-Cycle, the signals on the $\mathbf{T}$ half cycle can be delayed until the corresponding u_i and d_i data units are received.

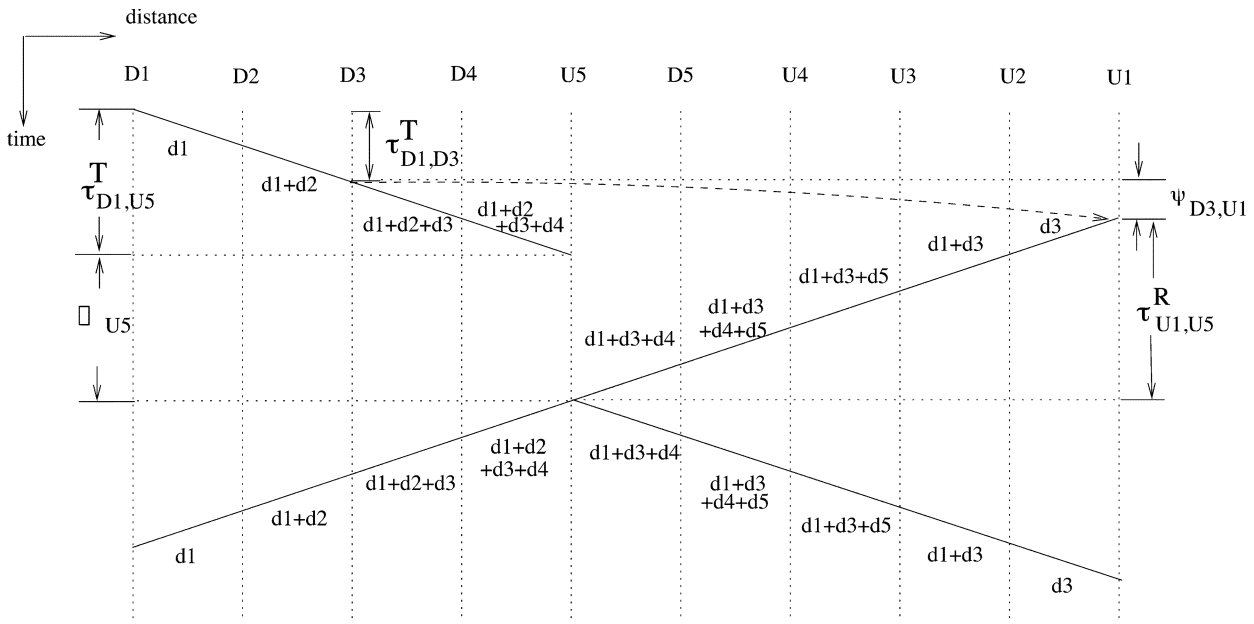

Fig. 5. Example of the timing considerations, and delay at U_k nodes ($U_k = U_5$ in this example).

connection between nodes D_2 and U_5 has failed, i.e., $D_i = D_2$ and $U_k = U_5$.

The derivation is based on the following assumptions:

- ψ_{D_j,U_1} is the delay over the working path from node D_j to node U_1.
- $\tau^R_{U_1,U_k}$ is the delay between U_1 and U_k on the $\mathbf{R}$ cycle.
- $\tau^T_{D_1,U_k}$ is the delay between D_1 and U_k on the $\mathbf{T}$ cycle, and similarly is $\tau^T_{D_1,D_j}$.
- Since shortest path communication is used, the propagation delay between a pair of communicating nodes over the primary path is always shorter than that over either the $\mathbf{T}$ or the $\mathbf{R}$ cycles.
- Node D_j is the one connected to node U_1.
- Assume that a node transmits its data unit on the working circuit and the $\mathbf{T}$ cycle at the same time.

Due to the last assumption, for a round to start on cycle $\mathbf{R}$, a delay of $\tau^T_{D_1,D_j} + \psi_{D_j,U_1}$ is required. This is shown in Fig. 5, which is the space-time diagram corresponding to the example in Fig. 4. In this example, node D_j is D_3. Hence, we have

$$\begin{aligned}\text{Delay at } U_k &= \Delta_{U_k} \\ &= \tau^T_{D_1,D_j} + \psi_{D_j,U_1} + \tau^R_{U_1,U_k} - \tau^T_{D_1,U_k}. \quad (3)\end{aligned}$$

The first two terms in the above equation correspond to the time from the start of a round on cycle $\mathbf{T}$ to the start of the same round on cycle $\mathbf{R}$, as stated above. Then, we add to it the time for this round's data to arrive at U_k on the $\mathbf{R}$ cycle (the third term). Finally, the time for the data in the same round to arrive at U_k on the $\mathbf{T}$ cycle is subtracted (the last term). By the choice of D_j, then

$$\tau^T_{D_1,D_j} - \tau^T_{D_1,U_k} \le 0. \quad (4)$$

Also

$$\psi_{D_j,U_1} + \tau^R_{U_1,U_k} \le \tau. \quad (5)$$

This inequality is valid since if it was not, then D_j and U_1 will use $\tau^R_{D_j,U_1}$ as it will be shorter.

Using (4) and (5) in (3) we obtain

$$\Delta_{U_k} \le \tau.$$

In the example in Fig. 4, this delay is introduced at node U_5, assuming that the working circuit between nodes D_2 and U_5 in Fig. 4 has failed. MSPP devices which can accommodate a 128 ms differential delay, can support this implementation.

Using the same method above, we obtain an upper bound on 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. Using D_1 as a references, the outage time at node U_k, Θ_{U_k}, is given by

$$\begin{aligned}\Theta_{U_k} = &\left(\tau^T_{D_1,D_j} + \psi_{D_j,U_1} + \tau^R_{U_1,U_k}\right) \\ &- \left(\tau^T_{D_1,D_i} + \psi_{D_i,U_k}\right). \quad (6)\end{aligned}$$

The derivation of the above equation is similar to that of (3), except that we subtract the time from the beginning of the round to the reception of d_i by node U_k (the last term).

Since any working path is shorter than $\tau/2$, and since

$$\tau^T_{D_1,D_j} + \tau^R_{U_1,U_k} < \tau$$

where we used the assumption of symmetry between the $\mathbf{T}$ and $\mathbf{R}$ cycles, then we have

$$\Theta_{U_k} < 1.5\tau.$$

If the last assumption above is relaxed, and all nodes are synchronized to transmit on the working paths at the same time, e.g., using a network clock, then the first and fourth terms in (6) will disappear, and the delay will become

$$\Theta'_{U_k} = \left(\psi_{D_j,U_1} + \tau^R_{U_1,U_k}\right) - \psi_{D_i,U_k} \quad (7)$$

which still has a loose upper bound of 1.5τ. In order to reduce the upper bound, and provide tighter guarantees on the outage time, all sources can start transmitting simultaneously, and at the same time both the $\mathbf{T}$ and $\mathbf{R}$ cycles can start. In order to make sure that the transmissions on the cycles will include valid data units, initially nodes are assumed to generate zero data units, which are not transmitted on the working paths, but are assumed to be received by the receivers. The number of such data units are those transmitted within a duration of $\max_{i,j} \psi_{D_i,U_j}$. In this case, the outage time will be given by

$$\Theta''_{U_k} = \max\left(\tau^R_{U_1,U_k}, \tau^T_{D_1,U_k}\right) - \psi_{D_i,U_k} \tag{8}$$

which is upper bounded by τ.

B. Synchronization

Since data units that are to be combined together must belong to the same round, then all data units of the same round must be present in order to form the linear combination that will be transmitted on the outgoing link of a cycle. This requires the use of a synchronization mechanism. However, synchronization can be easily implemented based on the adoption of two mechanisms, namely:

1) Round numbers, and
2) Buffers that will hold data units that are to be combined, including the input linear combinations.

The buffer, e.g., at node D_i which has a connection to node U_j, will be used to hold transmitted d_i data units, received u_j data units, and the linear combinations received on both $\mathbf{T}$ and $\mathbf{R}$ cycles. Once the data units belonging to the same round number are available at the head of their buffers, the output linear combination is formed and transmitted on the outgoing link.

C. Nodal Degree and p-Cycles

In order to implement the above scheme, each node should be able to transmit on three ports. If simple p-Cycles are used, then the implementation of this technique may not be feasible if source and/or destination nodes have a nodal degree of 2. However, since the on-cycle links are not protected, non-simple cycles may be used. In fact, the use of non-simple cycles may even result in lowering the protection requirements, since a non-simple cycle that traverses a set of connection end nodes may require a number of links which is less than that required by simple cycles.

D. Channel Errors

The proposed scheme is robust with respect to channel errors, especially those which affect the composite signal. That is, once the composite signal is hit by an error burst, the error can be detected and removed, and this will take place within no more than two hops (of connection end-nodes): one hop for detection, and a second hop for removal of the error. To see this, assume that an error burst hits the signal propagating on the $\mathbf{T}$ cycle just before it arrives at node D_i, which has a connection with node U_j. Let this error burst be represented by the polynomial E. Therefore, node D_i will receive $\hat{D}(Tx)^n_{i-1} + \hat{U}(Tx)^n_j + E$ on $\mathbf{T}$, and $\hat{U}(Rx)^n_{i+1} + \hat{D}(Rx)^n_j$ on $\mathbf{R}$. Let us consider two cases:

Case 1: No Failures:

In this case, the addition of the above two signals and the appropriate d_i and u_j signals will result in E. Detecting that E is nonzero indicates an error. Since node D_i does not know whether E has hit the signal received on $\mathbf{T}$ or the signal received on $\mathbf{R}$, it only detects the error, but does not remove it. Therefore, it sends a short signal (can be a single bit) to both neighboring nodes to indicate the possibility of an error. The downstream node on $\mathbf{T}$ from D_i will detect the error again, and because of the receipt of this signal, it can now remove the error by adding E to the $\mathbf{T}$ signal. The upstream node on $\mathbf{T}$ from D_i will not detect the error, and will therefore ignore the possible error indication signal received from D_i.

Case 2: A Working Path Failure:

In this case, node D_i will recover $u_j + E$. Node D_i can detect the presence of the error through the use of the CRC in the data unit. Notice that adding $u_j + E$ to the signal on $\mathbf{T}$ will remove both u_j and E. However, in the general case, since node D_i does not know which signal was hit by the error burst E, it will execute the same procedure in Case 1 by which it notifies its two neighboring nodes.

E. Implementation Notes

While this paper presents the theoretical bases of the proposed strategy, it is important to comment on the feasibility of implementing it. In fact, the proposed strategy can be implemented in a number of technologies and at a number of layers. For example, it can be implemented at layer 1 using NGS protocols, and in particular the GFP protocol. Since data units from different higher layer protocols are encapsulated in the payload field of GFP frames, the payload field can be used to accommodate the encoded (added) data units. It can also be implemented at layer 2 using ATM, where a special VCI/VPI can be reserved for a p-Cycle that protects a given set of VCCs or VPCs. The payloads of the ATM cells to be protected are therefore added and transmitted on the p-Cycle VCC. Moreover, it can be implemented at layer 3, and in particular using the IP protocol. With IP, the sum of data units (packets in this case) can be encapsulated in another IP packet. The encapsulating IP packet header would include the IP numbers (on two different interfaces) of the node that starts a round, e.g., D_1, as both the source and destination. Source routing may have to be used to make sure that this packet will traverse the p-Cycle. The strategy can also be implemented using MPLS where a certain LSP is used to implement the p-Cycle. All data units are added and the label of the p-Cycle LSP is prepended to the sum. In fact, with MPLS we have the advantage that paths are precomputed, and do not change during operation.

Note that the proposed strategy requires four mechanisms:

1) Data units are fixed and equal in size;
2) Round numbers can be indicated in data units;
3) There is an XOR addition mechanism at each node; and
4) There is a buffer equal to the round trip delay around the p-Cycle at each node.

The last two mechanisms are not difficult to provision.

In order to implement the first mechanism, and if data units cannot be made fixed in size, e.g., under IP, a number of ways can be used to circumvent this problem. One option would be that each node would concatenate (or block) its own data units and then segment them into fixed size segments. Another option would be to use data units of a length that is equal to the data unit with the largest size. Shorter data units are extended by adding trailing zeroes. The first option requires some processing, but is efficient in terms of bandwidth utilization. The second option, which is also feasible under a number of technologies, can lead to bandwidth degradation since the bandwidth reserved for protection in this case will be based on the maximum size data units. However, since it does not require blocking and segmentation, its processing requirements are less than those of the first option.

Providing round number can be also accommodated in a number of technologies. For example, when using GFP, a new extension header can be defined to include the round sequence number. With IP, the sequence number of the encapsulating IP header can act as the sequence number. It is to be noted that data units combined on the $\mathbf{T}$ and $\mathbf{R}$ cycles can be offset by a number of rounds that depends on a, where a is given by

$$a = \left\lceil \frac{\tau}{(\text{Protection data unit size in bits})/B} \right\rceil. \quad (9)$$

τ in the above equation is the round trip propagation delay around the p-Cycle, and B is the protected transport capacity.

Considering the example of Fig. 4, and if rounds on the $\mathbf{T}$ cycle are started by node D_1, then at node D_3, if d_1 and d_2 belong to round n, then u_1, u_3 and u_4 belong to round $n-a$. Therefore, the indication of the round number must also indicate the round number of each data unit. This can be accommodated by including:

1) The round number n, and
2) For each data unit, whether it belongs to round number n, or to a preceding round number, as will be indicated below.

For this purpose, we propose a round number field that includes the *round number* and one round number bit for each data unit. This last bit will be the same as the least significant bit (LSB) in the *round number* if the data unit belongs to the same round number, n, or the complement of the LSB if the data unit belongs to an earlier round number, $n-a$ on $\mathbf{T}$, or $n-2a$ on $\mathbf{R}$. Assuming that a is odd,[6] then since a is added to the round number field to update round numbers by D_1, the round number bits for all nodes will contain the above information. Fig. 6 shows this field and how it is set for the combined data units received by node D_3 on the $\mathbf{T}$.

The determination of which data unit to add to round n on the $\mathbf{T}$ and $\mathbf{R}$ cycles is as explained below. Let us assume that node D_i (U_i) communicates with node $U_j(D_j)$.

- On $\mathbf{T}$: node D_i (U_i) complements its round bit. If the round bit of node U_j (D_j) is the same as that of the current node's round bit, it adds $d_i(n)+u_j(n)$ $(u_i(n)+d_j(n))$ to $\mathbf{T}$. Otherwise, it adds $d_i(n)+u_j(n-a)$ $(u_i(n)+d_j(n-a))$.
- On $\mathbf{R}$: node D_i (U_i) complements its round bit. If the round bit of node U_j (D_j) is the same as that of the current node's round bit, it adds $d_i(n)+u_j(n-a)$ $(u_i(n)+d_j(n-a))$ to $\mathbf{R}$. Otherwise, it adds $d_i(n)+u_j(n-2a)$ $(u_i(n)+d_j(n-2a))$.

By doing this, node D_i (U_i) will guarantee that the two combinations arriving on the $\mathbf{T}$ and $\mathbf{R}$ cycles in round n correspond to the same combinations.

round number	d_1	d_2	d_3	d_4	d_5	u_1	u_2	u_3	u_4	u_5
01101	1	1	0	0	0	0	0	0	0	0

Fig. 6. The round number field, and an example showing how it is set when received by node D_3 on the $\mathbf{T}$ cycle.

VII. Hybrid $1+\mathrm{N}$ and $1:\mathrm{N}$ Protection

Unlike p-Cycles used for $1:\mathrm{N}$ protection, the $1+\mathrm{N}$ protection scheme proposed in this paper does not protect circuits which share links with the p-Cycle, i.e., on-cycle links. The reason is due to the use of network coding on the p-Cycle, which means that if an on-cycle link fails, the cycle will be broken, and cannot be used to deliver the coded data to the end nodes of the failed link. However, the $1+\mathrm{N}$ protection scheme can be combined with the legacy p-Cycle $1:\mathrm{N}$ protection scheme to protect circuits sharing links with the p-Cycle. In case a working link on the p-Cycle fails, network coding will be disabled, and the the circuits sharing links with the p-Cycle can be rerouted on the p-Cycle, which requires end-node switch reconfiguration. This will result in reducing the overall protection redundancy. However, the failure recovery times for the $1:\mathrm{N}$ protected circuits are expected to exceed those which are protected by the $1+\mathrm{N}$ scheme. We refer to this strategy as a hybrid $1+\mathrm{N}$ and $1:\mathrm{N}$ protection. It should be noted that in the case when there are no straddling links, or circuits, this hybrid strategy becomes the $1:\mathrm{N}$ protection scheme.

In this section, we describe the basics of Hybrid $1+\mathrm{N}/1:\mathrm{N}$ protection scheme for link protection[7]. All of the previous operational assumptions still hold in this case. However, a p-Cycle will be provisioned to protect on-cycle and straddling links. In addition to the nodes in $\mathcal{D}$ and $\mathcal{T}$, which are at the ends of straddling links, we define the set of nodes $\mathcal{X}$. Node $X_i \in \mathcal{X}$ is not at the end of a straddling link, but is an end node of an on-cycle link only. We also define $\mathbf{C}(D_i)$ and $\bar{\mathbf{C}}(D_i)$ as the next node in the clockwise and counterclockwise directions on the p-Cycle from node D_i, respectively. Similarly, we define $\mathbf{C}(U_i)$, $\bar{\mathbf{C}}(U_i)$, $\mathbf{C}(X_i)$ and $\bar{\mathbf{C}}(X_i)$. We denote the data units sent in round n on the on-cycle working links by node D_i to nodes $\mathbf{C}(D_i)$ and $\bar{\mathbf{C}}(D_i)$ by $s_i^{\mathbf{C}}(n)$ and $s_i^{\bar{\mathbf{C}}}(n)$, respectively. Similarly, we define the data units sent by U_i and X_i in the clockwise and counterclockwise directions as $t_i^{\mathbf{C}}(n)$, $t_i^{\bar{\mathbf{C}}}(n)$, $x_i^{\mathbf{C}}(n)$ and $x_i^{\bar{\mathbf{C}}}(n)$, respectively.

A node on the p-Cycle can be one of two types:

[6] a is assumed to be odd for simplicity. If a is even, since node D_1 starts rounds, it will have to complement all bits in the round number bits for all nodes, by simply XORing a vector of 1's with such bits. However, a does not need to correspond to the actual value of a, but can be taken as the smallest odd integer greater than or equal to the value given by (9).

[7] The scheme works with either straddling links or straddling paths.

Type 1: An end node of both an on-cycle link, and a straddling link. Nodes in $\mathcal{D}$ and $\mathcal{U}$ are of this type. Or,

Type 2: An end node of an on-cycle link only. Nodes in $\mathcal{X}$ are of this type.

As described earlier, one of the Type 1 nodes will start rounds on the p-Cycle in both directions, $\mathbf{T}$ and $\mathbf{R}$, which will be used in exactly the same way described in Section V. Without loss of generality, let D_1 be the node to start the rounds: It will start round n on $\mathbf{T}$ by transmitting $d_1(n)$, and will start round n on $\mathbf{R}$ without transmissions. Node U_1 will be the first node to transmit on $\mathbf{R}$ in round n.

A Type 1 node will do two things:

1) It will behave similar to an on-cycle node in the $1+\mathrm{N}$ protection scheme described in Section V. The data units combined by the Type 1 nodes and transmitted on the p-Cycle are used to protect *straddling links*, are called *Straddling Links Protection* (SLP) data units.
2) If a Type 1 D_i node does not receive a data unit on the $\mathbf{T}$ cycle, it assumes that the link on the $\mathbf{T}$ cycle between $\bar{\mathbf{C}}(D_i)$ and D_i has failed, and sends the $s_i^{\bar{\mathbf{C}}}$ downstream on the $\mathbf{T}$ cycle (i.e., in a direction opposite to that of the working link) so that it can be received by node $\bar{\mathbf{C}}(D_i)$. Also, if the node does not receive a data unit on the $\mathbf{R}$ cycle, it assumes that the link on $\mathbf{R}$ cycle between $\mathbf{C}(D_i)$ and D_i has failed and sends $s_i^{\mathbf{C}}$ downstream on the $\mathbf{R}$ cycle, so that it can be received by $\mathbf{C}(D_i)$. In the above two cases, node D_i also receives the data units from $\bar{\mathbf{C}}(D_i)$ and $\mathbf{C}(D_i)$ on $\mathbf{R}$ and $\mathbf{T}$, respectively. Node U_i will behave similarly.
   The data units which are used to protect *on-cycle links* are called *On-Cycle Links Protection* (OLP) data units.

A Type 2 node, $X_i \in \mathcal{X}$, will only perform Step 2 performed by Type 1 nodes only, and will transmit OLP data units only.

Two more mechanisms are needed:

1) At any of the nodes on the cycle, SLP data units have transmission precedence on the cycle over OLP data units.
2) At the node that starts the cycles, D_1, SLP data units for round n are not generated unless SLP data units for round $n-a$ are received, where a is given in (9).

We show an example in Fig. 7 of a p-Cycle protecting five nodes, four of Type 1, D_1, D_2, U_1, and U_2, and one node of Type 2, X_1. In the absence of failures, the data units transmitted on the working links are shown in Fig. 7(a), while the linear combinations carried on the $\mathbf{T}$ and $\mathbf{R}$ cycles are shown in Fig. 7(b). When a straddling link fails, e.g., between D_1 and U_2 shown in Fig. 7(b), the combinations received at D_1 and U_2 can be used to recover u_2 and d_1, respectively. However, when an on-cycle link fails, e.g., between nodes X_1 and D_2, the $\mathbf{T}$ cycle is used to carry $s_2^{\bar{\mathbf{C}}}$ to X_1 in the clockwise direction. Similarly, the $\mathbf{R}$ cycle is used to carry $x_1^{\mathbf{C}}$ data units to D_2, and in the counterclockwise direction. This is shown in Fig. 7(c).

VIII. Extensions to Multipoint Communication

In this section we discuss how the proposed technique can be used to protect multipoint connections, *viz.*, one-to-many and many-to-one.

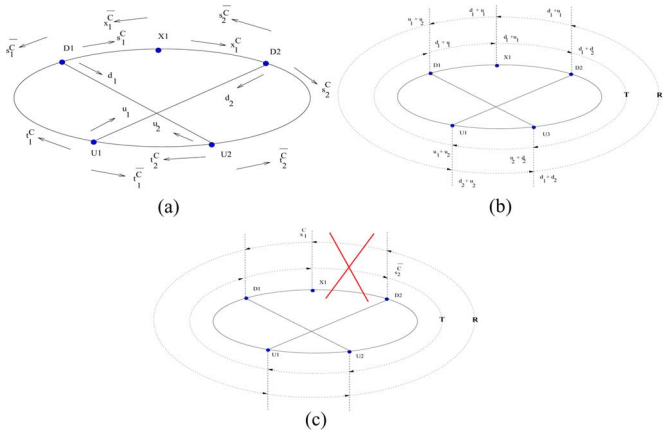

Fig. 7. An example of a p-Cycle used to protect 2 straddling and 5 on-cycle links. (a) The working links. (b) The protection circuits used to protect straddling links. (c) The protection circuits used to protect on-cycle links.

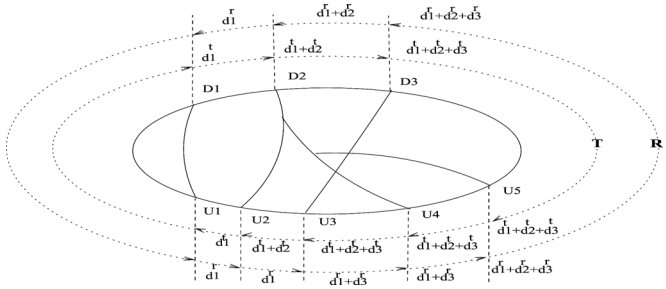

Fig. 8. $1+\mathrm{N}$ protection of multicast connections.

A. One-to-Many Sessions

We illustrate the procedure for handling one-to-many, or multicast, sessions by considering the case of the transmission of d_i units from node D_i in $\mathcal{D}$ to multiple destination nodes in $\mathcal{U}$. A similar procedure can be implemented for transmissions from a node on $\mathcal{U}$ to nodes in $\mathcal{D}$. We denote by U_c and U_f the destinations in the session that are, respectively, the closest and the farthest from the session source in $\mathcal{D}$ on the $\mathbf{T}$ cycle in the clockwise direction. These two nodes have the following responsibilities:

- Node U_c adds data units d_i to the $\mathbf{R}$ cycle. It does not act on the data received on the $\mathbf{T}$ cycle.
- Node U_f removes data units d_i from the $\mathbf{T}$ cycle. It does not act on the data received on the $\mathbf{R}$ cycle.

Based on the above, in the case of failure all destination nodes in the multicast session will receive $\sum_{j,D_j\in B, j\neq i} d_j + d_i$ on cycle $\mathbf{T}$, and $\sum_{j,D_j\in B, j\neq i} d_j$ on cycle $\mathbf{R}$, where B is a subset of $\mathcal{D}$. This enables such destinations to recover the d_i units in case of failure. This is shown in the example in Fig. 8 where D_2 transmits data units d_2 to U_2, U_4 and U_5.

The above may require buffering data on the $\mathbf{T}$ cycle at U_f until data in the corresponding round arrives from upstream on the $\mathbf{R}$ cycle. Or, it may require buffering data on the $\mathbf{R}$ cycle at U_c until data in the corresponding round arrives from upstream on the $\mathbf{T}$ cycle. Buffering at both nodes is not required.

Note that the above strategy can tolerate the failure of multiple links on the multicast tree from D_i to its destinations in $\mathcal{U}$.

B. Many-to-One Sessions

In the case of many-to-one sessions, the adaptation of the proposed strategy is straightforward. In this case, the destination node can be regarded as multiple destinations, and it applies the basic strategies m times, where m is the number of sources in the session. Since the destination node will be receiving at a rate that is equal to the sum of the rates of all transmitters, the p-Cycle must operate at least at this rate. Therefore, data units transmitted by the sources of the same session can be time multiplexed on the p-Cycle. The paths from the sources to the destination need not be link disjoint.

IX. Cost Evaluation of $1 + \mathrm{N}$ Protection

In this section we evaluate the cost of $1 + \mathrm{N}$ protection using p-Cycles, and compare it to the cost of $1+1$ protection. The cost evaluation of $1+1$ and $1+\mathrm{N}$ protection is based on optimal formulations. For the case of $1+1$ protection, Bhandari's algorithm [15] was used to find two links disjoint paths between the end nodes of each connection. An integer linear programming (ILP) formulation is used to evaluate the cost of $1+\mathrm{N}$ protection. The ILP formulation is given in the Appendix. These will be used to carry out an empirical comparison between the cost of implementing both strategies. It is to be noted that the cost metric used in this paper is the number of links, where a span may contain a number of links, e.g., DS-3 circuits or wavelengths.

We compare the cost of implementing $1+1$ and $1+\mathrm{N}$ protection strategies using random graphs, while assuming that there is no upper bound on the number of links per span. In our experiments, we allow the use of non-simple cycles. Therefore, and due to the complexity of the problem, we ran our experiments using 8-node networks. The networks were generated randomly such that each sample network contained a given number of edges, and that the network is at least bi-connected. For the generated network, we provisioned a given number of connections, such that the end points of the connections were uniformly selected from all the nodes in the network. For each experiment, we generated 10 sample networks, and calculated the average of the number of protection and working circuits over all the runs. In the examples below, we show the total number of wavelength links, and between parentheses we show the number of protection and working circuits, respectively.

In the first example, shown in the first half of Table I, the network has 8 nodes, and 12 edges. The average nodal degree in this case is 3. In the examples, we show the total network cost, and the cost of primary and protection paths. The table shows that $1+1$ protection performs better than $1+\mathrm{N}$ protection, both in terms of the number of working and protection circuits. Notice that when the number of connections is equal to the number of links in the graph (the case referred to as *links*), i.e., the case of link protection,[8] the number of working circuits is exactly the same in both cases, but the number of protection circuits is about 15% more in the case of $1 + \mathrm{N}$. That is, $1 + \mathrm{N}$ protection has no advantages in this case. However, as the network becomes denser, $1 + \mathrm{N}$ protection requires fewer circuits than $1 + 1$ protection. This is shown in the second half of Table I, where the nodal degree in this case is 4. Although the number of protection circuits exceeds the number of working circuits under $1+\mathrm{N}$ protection, but the cost of protection circuits under $1+\mathrm{N}$ protection is at least 30% lower than that under $1+1$ protection. In Table II we show the cost of $1 + 1$ and $1 + \mathrm{N}$ protection when link protection for all links in the network is provided. Four networks are considered, two six node networks, with 10 and 12 edges respectively, and two eight node networks, similar to those in Table I. In these examples, and similar to the conclusion drawn from the above two examples, it is shown that the cost of $1 + \mathrm{N}$ protection becomes less than the cost of $1 + 1$ protection as the network density increases. It is to be noted that there is a large number of networks with a high nodal degree, i.e., 4 or more. Examples of which include the NJ-LATA with a nodal degree of 4, and the Pan-European COST239 network with a nodal degree of 4.7. Such networks may be regarded as candidates for the use of the proposed strategy.

TABLE I
Comparison Between $1 + 1$ and $1 + \mathrm{N}$ Protection in an 8-Node Network

$\|E\|$	N	1+1			1+N		
		Total	Working	Spare	Total	Working	Spare
12	12 (links)	39	12	27	43	12	31
	10	41	16	25	50	24	26
	8	31	12	19	37	13	24
16	16 (links)	51	16	35	39	16	23
	14	49	19	36	45	20	25
	12	44	18	26	34	16	18

TABLE II
Full Link Protection

$\|V\|$	$\|E\|$	1+1			1+N		
		Total	Working	Spare	Total	Working	Spare
6	10	30	10	20	30	10	20
	12	36	12	24	26	12	14
8	12	39	12	27	43	12	31
	16	51	16	35	39	16	23

One thing that was observed from the above results is that the maximum number of links per span under $1 + \mathrm{N}$ protection is less than under $1+1$ protection. For example, for a network of 8 nodes and 12 edges, protecting 10 connections using $1 + 1$ protection required several spans to be provisioned with 5 links on the same span. With $1 + \mathrm{N}$ protection, however, only one span needed to be provisioned with 4 links, and the rest were provisioned with either 1 or 2 links. This means that restricting the number of links per span to a certain upper bound may change the cost significantly. This is the subject of future study. It is to be also noted that the saving introduced by $1 + \mathrm{N}$ protection over $1 + 1$ protection is somewhat limited in this study. This is due to the use of small networks, which were the only networks that the ILP was able to solve in reasonable time.

We also illustrate the cost of the Hybrid $1 + \mathrm{N}/1 : \mathrm{N}$ protection, and compare it to the cost of $1 + 1$ protection. The cost of the Hybrid $1+\mathrm{N}/1 : \mathrm{N}$ protection is based on using the ILP formulation which is similar to that in [16]. However, we modified the formulation in [16] in order to also maximize the number of links which are protected using $1 + \mathrm{N}$ protection, without resulting in increasing the number of protection circuits. Due to

[8]The connections were embedded in the ILP such that each of the $|E|$ connections is routed over exactly one edge in the network graph.

TABLE III
COMPARISON BETWEEN 1 + 1 AND HYBRID 1 + N PROTECTION

		1+1 Protection	Hybrid 1+N/1:N Protection	
$\|V\|$	$\|E\|$	protection cost	protection cost	# straddling links
	8	56	8	0
8	12	30	9	4
	16	32	8	8
	20	40	8	12
	14	182	14	0
14	21	65	16	6
	28	56	20	19
	35	70	15	24

the lack of space, we do not include the ILP formulation in the paper.

The experiments considered a number of networks where the number of nodes assumed two values, 8 and 14 nodes. We allowed the graph density for each network to assume one of four values, namely, 1, 1.5, 2, and 2.5. The graphs were generated randomly, but we made sure that all graphs were at least bi-connected. For each network, 8 different random graphs were generated, and we took the average of the results.

In Table III, we show the cost of the protection circuits required for both $1 + 1$ and Hybrid $1 + \mathrm{N}/1 : \mathrm{N}$ protection. For the Hybrid $1 + \mathrm{N}/1 : \mathrm{N}$ protection, the number of links which are protected as straddling links, i.e., using $1 + \mathrm{N}$ protection, is also shown.

Under $1 + 1$ protection, the worst case cost of protection circuits is always when the nodal degree is 2, i.e., the network has a ring topology. There is exactly one way of choosing the protection path, namely, the entire ring topology excluding the protected link. However, under Hybrid $1 + \mathrm{N}$ protection, the problem reduces to p-Cycle protection, where all the protected links are on-cycle links, and the cycle corresponds to the entire graph. This results in the largest percentage of protection circuits, 100%. Note that in this case, for the Hybrid $1 + \mathrm{N}$ protection, there are no $1 + \mathrm{N}$ protected links, and it is $1 : \mathrm{N}$ protection. As the number of edges increases, and consequently the nodal degrees, the cost of $1 + 1$ protection remains high, which is always around 200% of the cost of working links. Under Hybrid $1 + \mathrm{N}$ protection, the ratio of the protection circuits to the working circuits decreases. Notice also that as the number of edges increases, the number of links which are $1 + \mathrm{N}$ protected, i.e., straddling links, also increases. For example, with a graph density of 4, at least 50% of the links are protected using $1 + \mathrm{N}$ protection, since they are straddling links. The remaining links are $1 : \mathrm{N}$ protected.

X. CONCLUSION

This paper presented the principles of a scheme for achieving $1 + \mathrm{N}$ 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. It also simplifies the management and control planes, and can also provide a mechanism for error detection and correction. The scheme can be implemented at a number of layers and using a number of protocols including IP, or GFP in NGS. In order to protect on-cycle links, a hybrid $1 + \mathrm{N}/1 : \mathrm{N}$ strategy was presented in which on-cycle links are protected using $1 : \mathrm{N}$ protection. A performance evaluation study showed that as the density of the graph increases the efficiency of the proposed $1 + \mathrm{N}$ protection scheme improves in terms of decreasing the ratio of the required protection circuits compared to the working circuits. Moreover, the $1 + \mathrm{N}$ protection becomes more efficient than $1 + 1$ protection under the same conditions. Therefore, the proposed strategy can be a candidate for use in networks with high average nodal degrees, such as NJ-LATA and the Pan-Eurpoean COST239 networks.

Future work will consider implementing the proposed strategy in different technologies. It will also consider detailed performance evaluation and the effect of practical network dimensioning limitations on the performance. Moreover, since the ILP formulations presented in this paper have a high complexity, which limits their applicability to small networks, we plan to develop heuristic and approximate approaches, which may be suboptimal, but may allow provisioning of larger numbers and a greater number of connections.

APPENDIX
ILP FOR THE MINIMAL COST 1 + N PROTECTION

This Appendix finds the minimal cost provisioning for $1 + \mathrm{N}$ protection in mesh networks using an ILP formulation. The cost is defined in terms of the number of wavelength links. It is assumed that the number of wavelength channels is not upper bounded. It is also assumed that wavelength conversion is available at all nodes, and therefore wavelength continuity is not enforced. In this case, several copies of the same p-Cycle may be needed. This is because two connections may have to be protected by the same cycle, but such connections cannot be routed on link disjoint paths. Multiple copies of the same p-Cycle must be routed on different wavelength channels, or different circuits, depending on the unit of protection. We start by finding the set of all cycles $\mathbb{P}$ in the network graph. The ILP formulation assumes that there are K copies of each cycle.

The following table defines the parameters and variables used in the formulation:

N	number of connections (input parameters)
$s(k)$	source of connection k (input parameter)
$d(k)$	destination of connection k (input parameter)
Z_{ij}^{k}	binary variable which is 1 if and only if connection k uses link (i, j) on primary path
$\mathbb{P}_k$	subset of cycles that may be used to protect connection k, i.e., $s(k)$ and $d(k)$ are on a cycle in $\mathbb{P}_k$
L^p	length of cycle p (input parameter)
$C_{i,j}^{p}$	binary variable which is 1 if link (i, j) is on cycle p (input parameter)
K	number of copies of each cycle (input parameter)
$\delta_k^{p,m}$	binary variable which is 1 if copy m of cycle p is used by connection k
$\Delta^{p,m}$	binary variable which is 1 if copy m of cycle p is used

Minimize:

$$\sum_{k,i,j} Z_{i,j}^{k} + \sum_{p,m} \Delta^{p,m} L^{p}$$

Subject to:

$$Z_{is}^{k} = 0 \quad \forall k, i \neq s(k) \tag{10}$$

$$Z_{dj}^{k} = 0 \quad \forall k, j \neq d(k) \tag{11}$$

$$\sum_{i \neq s(k)} Z_{si}^{k} = 1 \quad \forall k \tag{12}$$

$$\sum_{i \neq d(k)} Z_{id}^{k} = 1 \quad \forall k \tag{13}$$

$$\sum_{i} Z_{ij}^{k} = \sum_{i} Z_{ji}^{k} \quad \forall k, j \neq s(k), d(k) \tag{14}$$

$$Z_{ij}^{k} + Z_{ji}^{k} + \delta_{k}^{p,m} \left(C_{ij}^{p} + C_{ji}^{p} \right) \leq 1 \quad \forall i, j, k, p \in \mathbb{P}_k, 1 \leq m \leq K \tag{15}$$

$$\delta_{k}^{p,m} = 0 \quad \forall k, p \notin \mathbb{P}_k, 1 \leq m \leq K \tag{16}$$

$$\sum_{m,p \in \mathbb{P}_k} \delta_{k}^{p,m} = 1 \quad \forall k \tag{17}$$

$$\delta_{k}^{p,m} + Z_{ij}^{k} + Z_{ji}^{k} + \delta_{l}^{p,m} + Z_{ij}^{l} + Z_{ji}^{l} \leq 3 \quad \forall k, l, k \neq l, i, j, p, 1 \leq m \leq k \tag{18}$$

$$\frac{\sum_{k,\ \mathrm{s.t.}\ p \in \mathbb{P}_k} \delta_{k}^{p,m}}{Q} \leq \Delta^{p,m} \leq \sum_{k, \mathrm{s.t.}\ p \in \mathbb{P}_k} \delta_{k}^{p,m} \quad \forall p, 1 \leq m \leq K \tag{19}$$

Equations (10), (11), (12), (13), and (14) ensure that the traffic on the working path is generated and consumed by the source and destination nodes, respectively. Equation (15) makes sure that connection k and the cycle used to protect k are link disjoint. Equations (16) and (17) guarantee that there is only one cycle that will protect connection k, and this cycle is one of the cycles in $\mathbb{P}_k$. Equation (18) ensures that if two connections share a link and are protected by the same cycle, they are protected by different copies of the same cycle. The number of copies of each cycle is evaluated in (19).

REFERENCES

[1] D. Zhou and S. Subramaniam, "Survivability in optical networks," *IEEE Network*, vol. 14, no. 6, pp. 16–23, Nov./Dec. 2000.

[2] D. Stamatelakis and W. D. Grover, "Theoretical underpinnings for the efficiency of restorable networks using preconfigured cycles (p-cycles)," *IEEE Trans. Commun.*, vol. 48, no. 8, pp. 1262–1265, Aug. 2000.

[3] D. Stamatelakis and W. D. Grover, "Ip layer restoration and network planning based on virtual protection cycles," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 10, pp. 1938–1949, Oct. 2000.

[4] W. D. Grover, *Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking.*. Upper Saddle River, NJ: Prentice-Hall, 2004.

[5] G. Shen and W. D. Grover, "Extending the p-cycle concept to path segment protection for span and node failure recovery," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 10, pp. 1306–1319, Oct. 2003.

[6] A. Kodian and W. D. Grover, "Failure-independent path-protecting p-cycles: Efficient and simple fully preconnected optical-path protection," *J. Lightw. Technol.*, vol. 23, no. 10, pp. 3241–3259, Oct. 2005.

[7] W. D. Grover and A. Kodian, "Failure-independent path protection with p-cycles: Efficient, fast and simple protection for transparent optical networks," in *Proc. 7th ICTON*, Jul. 2005, pp. 363–369.

[8] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[9] T. Ho, D. R. Karger, M. Medard, and R. Koetter, "Network coding from a network flow perspective," in *Proc. Int. Symp. Inf. Theory*, 2003, p. 441.

[10] D. S. Lun, M. Médard, T. Ho, and R. Koetter, "Network coding with a cost criterion," in *Proc. Int. Symp. Inf. Theory and Its Appl.*, Parma, Italy, Oct. 2004, pp. 1232–1237.

[11] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2005.

[12] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[13] C. Fragouli, J.-Y. LeBoudec, and J. Widmer, "Network coding: An instant primer," *Comput. Commun. Rev.*, vol. 36, no. 1, pp. 63–68, Jan. 2006.

[14] E. Hernandez-Valencia, M. Scholten, and Z. Zhu, "The generic framing procedure (gfp): An overview," *IEEE Commun.*, vol. 40, no. 5, pp. 63–71, May 2002.

[15] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing.*. New York: Springer, 1999.

[16] W. He, J. Fang, and A. Somani, "A p-cycle based survivable design for dynamic traffic in WDM networks," presented at the IEEE Globecom, 2005.

Ahmed E. Kamal (S'82–M'87–SM'91) received the B.Sc. (distinction with honors) and the M.Sc. degrees in electrical engineering from Cairo University, Cairo, Egypt, in 1978 and 1980, respectively, and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of Toronto, Toronto, ON, Canada, in 1982 and 1986, respectively.

He is currently a Professor of Electrical and Computer Engineering at Iowa State University, Ames. Earlier, he held faculty positions with the Department of Computing Science, University of Alberta, Edmonton, Canada, and the Department of Computer Engineering, Kuwait University, Kuwait City, Kuwait. He was also an Adjunct Professor with the Telecommunications Research Labs, Edmonton, AB, Canada. His research interests include high-performance networks, optical networks, wireless and sensor networks, and performance evaluation.

Dr. Kamal is a Member of the Association for Computing Machinery and is a registered Professional Engineer. He was the co-recipient of the 1993 IEE Hartree Premium for papers published in *Computers and Control* in *IEE Proceedings* for his paper titled "Study of the Behaviour of Hubnet," and the Best Paper Award of the IEEE Globecom 2008 Symposium on Ad Hoc and Sensors Networks. He served on the Technical Program Committees of numerous conferences and workshops, was the organizer and Co-Chair of the first and second Workshops on Traffic Grooming 2004 and 2005, respectively, and was the Co-Chair of the Technical Program Committees of a number of conferences including the Communications Services Research (CNSR) Conference 2006, the Optical Symposium of Broadnets 2006, and the Optical Networks and Systems Symposium of the IEEE Globecom 2007, the 2008 ACS/IEEE International Conference on Computer Systems and Applications (AICCSA-08), and the ACM International Conference on Information Science, Technology and Applications, 2009. He is also the Technical Program Co-Chair of the Optical Networks and Systems Symposium of the IEEE Globecom 2010. He is on the editorial boards of the *Computer Networks* and the *Journal of Communications*.