New Diversity Coding Design Algorithms for Link Failure Recovery in Communication Networks

Serhat Nazim Avci *Student Member, IEEE* and Ender Ayanoglu *Fellow, IEEE*
Center for Pervasive Communications and Computing
Department of Electrical Engineering and Computer Science
University of California, Irvine
Irvine, CA 92697-2625

***Abstract*—Diversity coding is a form of network coding for link failure recovery in communication networks. Since it employs coding, there is no feedback signaling, and that feature makes it very fast. Previously, we have employed this basic technique and linear programming to come up with fast link failure recovery systems that also have small extra capacity. One approach, Diversity Coding Tree, employs mixed integer programming and results in very fast restoration. Another approach is called Coded Path Protection, and employs integer linear programming and has the advantage of small extra capacity. This latter technique is based on former work that considers a communication network as consisting of bidirectional links. However, employing coding on bidirectional links results in larger restoration times than the former technique. In this paper, we develop an improved version of our former techniques. This new technique employs a mixed integer linear programming formulation and results in restoration times as fast as Diversity Coding Tree with reduced extra capacity.**

I. Introduction

Incorporating network coding into link failure protection enables proactive protection. This results in much lower restoration time, lower signaling complexity, and higher transmission integrity compared to the rerouting-based techniques. This is a form of network coding and it was proposed in [1] and [2], prior to the first papers on network coding [3]. The technique is called *diversity coding,* and in its simplest form, N primary links are protected using a separate $N+1^{st}$ protection link which carries the modulo-2 sum, or XOR combination, of the data signals in each of the primary links.

In [4], [5], we developed a design algorithm for restoration via diversity coding in a network with arbitrary topology. In [4], restoration time advantage of diversity coding was shown over a path-based restoration technique [6] and *p*-cycle protection [7]. *P*-cycle protection offers fast recovery by simply employing pre-cross-connected protection cycles. A comparison of different pre-cross-connected schemes can be found in [8]. The technique in [9] outperforms *p*-cycle protection in both recovery speed and capacity efficiency. The idea of converting a Shared Path Protection (SPP) [10] solution to a coding-based solution was introduced in [11] and named Coded Path Protection (CPP). Simulation results validated the potential of this idea. In [12], [13], optimal design algorithms were developed for provisioning of the static and dynamic traffic. In [12], it was shown that sub-ms recovery can be achieved via diversity coding with proper synchronization and buffering, if connections share the same destination node. In [14], the basic structure of diversity coding was extended to incorporate both primary and protection paths in coding operations, resulting in improvement in capacity efficiency.

In [15], network coding was combined with *p*-cycle protection to offer fast recovery from link failures. The technique is called 1+N protection. It forms circular bidirectional protection paths to protect a group of bidirectional connection demands from single link failures. The encoding and decoding operations are carried out at the end nodes of the connection demands. The primary paths of the connection demands are link-disjoint with each other and with the protection cycles. The term link-disjointness actually refer to span-disjointness. We prefer the former term since it is more commonly used.

In [16], a new tree-based protection scheme was introduced and called Generalized 1+N protection (G1+N). When link transmission is bidirectional, the parity data over the protection path are produced by summing the primary data in both directions, symmetrically. This enables decoding of the parity signals without extra links between the destination nodes and the decoding node. As a result, G1+N becomes comparable with SPP in terms of capacity efficiency. Protection of N parallel unidirectional links via coding is shown in Fig. 1(a) [1], [2], and protection of N parallel bidirectional links via coding is shown in Fig. 1(b) [15], [16]. G1+N technique has higher capacity efficiency compared to the classical diversity coding scenario. However, it lacks the restoration speed of diversity coding. In [17], a new trail-based protection scheme is proposed. Failed data are recovered via a linear coded protection circuit. This structure is a modified version of the scheme in [15], resulting in higher capacity efficiency by moving from cyclic to linear protection topology.

In this paper, the Diversity Coding Tree algorithm is adapted to implement the general diversity coding structures without the need of a common destination node. This feature enables a deeper exploration of the connectivity inside the network, with the outcome of higher capacity efficiency.

This work is partially supported by NSF under Grant No. 0917176.

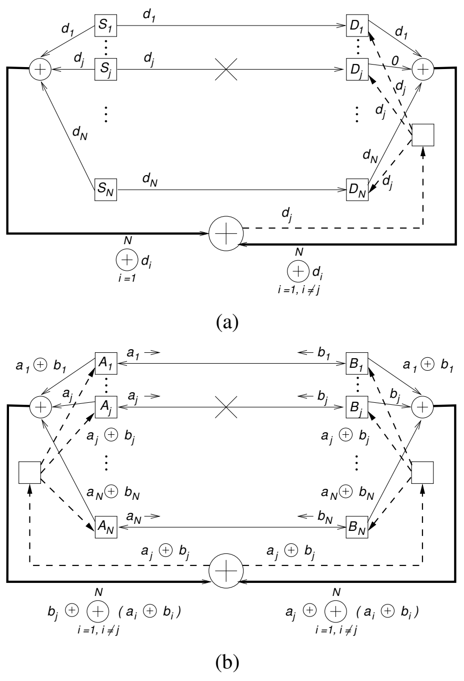
Fig. 1. Multipoint-to-multipoint architectures with coded protection against single link failures. (a) Unidirectional network [1], [2], and (b) bidirectional network [15], [16].

II. Coding Structure

The general diversity coding structure is shown in Fig. 1(a) for unidirectional connection demands. It has lower restoration time and higher spare capacity requirement compared to the G1+N protection technique shown in Fig. 1(b). G1+N protection is slower because of two different delays. First, the distance between the decoding node and the destination nodes are always higher than half of the propagation delay over the coding tree. Second, the parity data are transmitted only after a certain delay until the primary data, transmitted over the primary path, are received by the destination node. The extra links between the decoding node and the destination nodes in Fig. 1(a) makes the unidirectional scheme less capacity efficient than the bidirectional scheme in Fig. 1(b). In [12], we introduced the Diversity Coding Tree algorithm for optimal design of a coding-based protection, employed for links with a common destination node, resulting in fast restoration. This algorithm has a simpler coding structure than the one in Fig. 1(a) since the decoding operations are carried out at the single destination node. There is no need for feedback links or a designated decoding node. This coding structure results in very fast (sub-ms) restoration time. On the other hand, in [11], we introduced the Coded Path Protection (CPP) algorithm by optimally converting an SPP structure into a coding structure, resulting in high network capacity efficiency. This conversion results in smaller restoration time by eliminating most of the dynamic operations after the failure. As a side advantage, this algorithm has low complexity because of primary and protection paths are already available from the SPP solution. CPP has competitive capacity efficiency with respect to SPP. However, both Diversity Coding Tree and CPP have some potential for improvement. First, Diversity Coding Tree algorithm enforces a limited protection topology with connection demands going to the same destination node, eliminating some potentially capacity efficient coding group combinations. Second, CPP works on very flexible coding structures, but it lacks the restoration speed due to the bidirectional nature of the technique. As a result, in this paper, we developed Mixed Integer Programming (MIP) formulations to map the general unidirectional diversity coding structure to achieve both high capacity efficiency and low restoration time. Theoretically, the increased coding flexibility in the general diversity coding makes it more capacity efficient than diversity coding with single destination node. However, increased coding flexibility incurs increased design complexity which result in an extended time to reach close to optimal solutions.

As seen in Fig. 1(a), general diversity coding structure consist of N unidirectional connection demands from S_i to D_i for $1 \leq i \leq N$. Each connection demand has a link-disjoint primary path carrying data d_i. Then, the protection tree is the union of the protection paths that are protected. The protection tree is also link-disjoint to the primary paths. Each protection topology has a decoding node where decoding operation is carried out, which is shown by a big $\bigoplus$ sign. In addition to the protection tree, there is a feedback tree originating from destination nodes to the decoding node to help with decoding. The feedback tree overlaps with the protection tree, but transports data in the reverse direction. In most of the practical scenarios, the decoding node is very close to the destination nodes or is one of them. This minimizes the extra capacity due to the feedback tree. Moreover, the feedback tree becomes an empty set if there is a single destination node since decoding is carried out at that node. In the sequel, we will present the design algorithms to map the general diversity coding structure over arbitrary networks in the ascending order of complexity and optimality.

III. ILP Formulations

In this section, three design algorithms for a general diversity coding approach are presented with varying performance versus complexity tradeoffs. They are inspired by the "cycle-exclusion" technique in [18], which helps to achieve a simpler formulation then the algorithm in [16].

A. Coding Group Formation

The first MIP formulation is the Coding Group Formation (CGF) algorithm, which is the most basic version of design algorithms. In this algorithm, both the primary and protection paths are precalculated. Therefore, the algorithm simply minimizes the total capacity by optimally combining different connection demands, whose primary paths are link-disjoint. Each coding group has a protection tree and a feedback tree. The protection tree is the union of the protection paths of the connections in the same coding group. The feedback tree is the structure that ensures the decoding in a coding group. The complexity of this algorithm is significantly lower than the Diversity Coding Tree algorithm in [12] at the expense of optimality because the primary and protection paths are precalculated. The MIP formulation has the following parameters

- $G(V, E)$: Network graph,
- S : The set of spans in the network, a span consists of two links in opposite directions,
- N : Enumerated list of all unit-demand unidirectional connections,
- a_e : Cost associated with link e,
- T : Maximum number of diversity coding trees allowed, about half of the number of connections in each subproblem,
- $\Gamma_i(v)$: The set of incoming links of each node v,
- $\Gamma_o(v)$: The set of outgoing links of each node v,
- α : A constant employed in the algorithm, chosen very small,
- β : A constant employed in the algorithm, chosen very large,
- t : Diversity coding group index,
- i : Connection demand index,
- s_i : Source node of the connection demand i,
- d_i : Destination node of the connection demand i.

In addition to these parameters, in this version, the primary and protection paths of the connections are precalculated using 2-shortest link-disjoint paths routing algorithm in [7, p. 196] and input to the design algorithm as $[0, 1]$ binary parameters. The additional set of parameters are

- $y_e(i)$: Equals 1 iff the protection path of connection i traverses over span e, is acquired from the shortest path routing,
- $m(i, j)$: Equals 1 iff the primary path of connection i is link-disjoint to the primary and protection paths of connection j, is acquired from the shortest path routing.

Primary paths of the connections are not explicitly required in this formulation. Instead, the information about their link-disjointness is necessary and sufficient.

Next we provide the variables. Except the last one, they are binary and take the value of 0 or 1.

- $n(i, t)$: Equals 1 iff connection i is protected by the diversity coding group t,
- $c_e(t)$: Equals 1 iff the protection tree of coding group t passes through link e,
- $s_v(t)$: Equals 1 iff node v is a decoding node on protection tree t,
- $r_e(t)$: Equals 1 iff the feedback tree of coding group t passes through link e,
- $p_v(t)$: A continuous variable between 0 and 1, resulting in an MIP formulation. It keeps the "voltage" value of node v in the protection tree of t. It is possible to set this variable as an integer larger than 0 but that makes the simulation slower.

The first equation guarantees that a connection can be protected with only one diversity coding group

$$\sum_{t=1}^{T} n(i, t) = 1 \qquad \forall i \in N. \tag{1}$$

The following inequality makes sure that two connections can be in the same coding group if and only if the primary path of connection i is link-disjoint to the primary and protection paths of connection j and vice versa as

$$\begin{aligned} n(i, t) + n(j, t) \leq 1 + m(i, j),& \\ \forall i, j \in N, i < j, \forall t.& \end{aligned} \tag{2}$$

$$\begin{aligned} n(i, t) + y_e(i) \leq c_e(t) + 1,& \\ \forall i \in N, \forall e \in E, \forall t.& \end{aligned} \tag{3}$$

Inequality (3) ensures that link e is a part of the protection tree of coding group t if the protection path of connection i, which is in coding group t, passes through e. This inequality stems from the fact that the protection tree of a coding group is the union of the protection paths of the connections in the same coding group.

In inequality (4), the decoding nodes for each coding group t are identified. Unlike previously, in this formulation, the number of decoding nodes is unlimited for design purposes. The aim is to find the unique decoding node which is closest to the destination nodes to minimize the cost incurred by the feedback tree. It is also ensured that the root of the feedback tree is that decoding node. There are two cases to identify a decoding node. First, the protection tree is split into two or more outgoing links at a node. Second, the node has an outgoing link or links of protection tree and it is the destination node of some connections in the same coding group. The mathematical expression is

$$s_v(t) \geq \sum_{e \in \Gamma_o(v)} c_e(t) + \frac{\sum_{i=1, d_i=v} n(i, t)}{\beta} - 1 \qquad \forall v \in V, \forall t. \tag{4}$$

The following inequalities find the feedback tree to ensure decodability inside the network. The feedback tree of each coding group lies in the same span with the protection tree, but in the opposite direction. This can be expressed by

$$r_e(t) \leq c_f(t) \qquad \forall e, f \in g, e \neq f, \forall g \in S, \forall t. \tag{5}$$

Inequalities (6) and (7) dictate the behavior of the feedback tree both on the decoding nodes and on the non-decoding nodes, respectively as

$$\sum_{e \in \Gamma_i(v)} r_e(t) + 1 \geq \sum_{e \in \Gamma_o(v)} c_e(t) + s_v(t) \qquad \forall v \in V, \forall t, \tag{6}$$

$$\begin{aligned} \sum_{e \in \Gamma_i(v)} r_e(t) + \beta s_v(t) \geq \sum_{e \in \Gamma_o(v)} c_e(t) + \frac{\sum_{e \in \Gamma_o(v)} r_e(t)}{\beta} - 1& \\ \forall v \in V, \forall t.& \end{aligned} \tag{7}$$

The reason behind inequality (6) is the requirement of a feedback link as an input to each decoding node. In inequality (7), the intermediate nodes link the feedback tree from outgoing links to incoming links in the reverse direction.

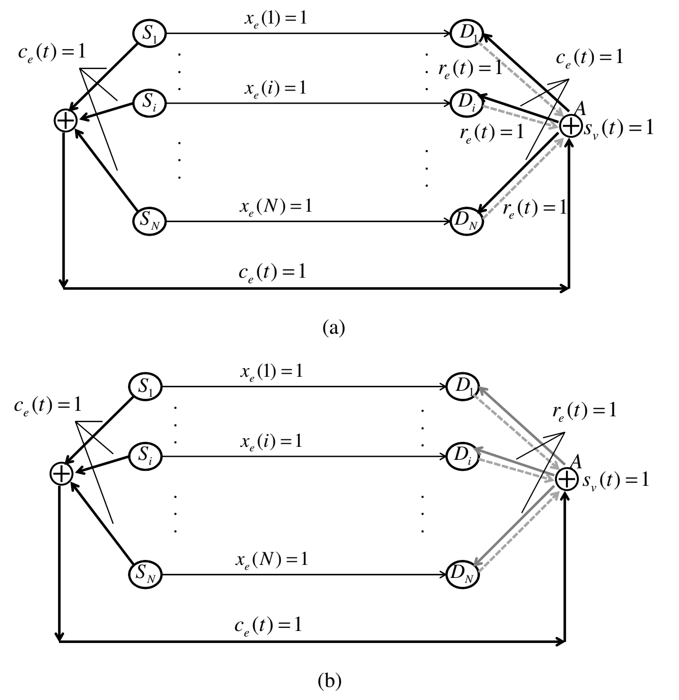

Fig. 2. MIP variables are set for different algorithms. (a) The CGF algorithm, and (b) the SCP algorithm.

To ensure that protection trees do not include any cyclic structures, we adopted the "voltage value" concept from [18]. In this formulation, the voltage value at the head node should be higher than the voltage value at the tail node of the links which are part of the protection tree.

The mathematical formulation of preventing cycles inside the protection tree is

$$p_v(t) - p_u(t) \geq \alpha \cdot c_e(t) - (1 - c_e(t)) \qquad \forall e = u \rightarrow v, \forall t. \quad (8)$$

The objective function is

$$\min \sum_{e \in E} \sum_{t=1}^{T} a_e \times (c_e(t) + r_e(t)). \qquad (9)$$

A typical diversity coding structure matched with MIP variables is depicted in Fig. 2(a). Thin straight arrows between source and destination nodes represent the primary paths, as $x_e(i)$ variables are set to 1. Thick straight arrows represent the protection tree. Therefore, the protection tree variable $c_e(t)$ is set to 1 over the protection tree. The decoding node variable $s_v(t)$ is set to 1 at the decoding node A. The dashed gray arrows represent the feedback tree between the destination nodes and the decoding node. The feedback tree variable $r_e(t)$ takes the value 1 over the feedback tree.

B. Spare Capacity Placement

The second design algorithm is the Spare Capacity Placement (SCP) algorithm. We have developed an MIP formulation to find the optimal spare capacity allocation and coding group combination given the primary paths. The primary paths are precalculated via the shortest path routing algorithm and input to the MIP formulation. The SCP algorithm has higher complexity and higher optimality compared to the algorithm in the previous section since the protection paths are calculated jointly with the coding group combination.

The novel approach is to redesign the MIP formulation without incurring extra variables and constraints on top of the CGF algorithm described in the previous section. To achieve that some significant changes are required on the formulation structure even if the underlying decoding structure is the same. First, in this formulation, there is only one specific decoding node, in contrast to the CGF algorithm. Second, the protection tree is divided into two link-disjoint trees. The tree between the source nodes and the decoding node is still called the protection tree. However, the tree between the decoding node and the destination nodes is combined with the feedback tree. These structures are shown in Fig. 2(b). The primary paths are the same as in the CGF algorithm. However, the protection tree shown by thick straight arrows terminates at the decoding node. This part of the tree is characterized as if these flows are destined to the decoding node. The part between the decoding node and the destination nodes is shown via gray lines. The thick straight gray arrows are the continuation of the protection tree. The dashed gray arrows are the feedback links. It is noted that feedback links and the second part of the protection tree share the same spans. To simplify the MIP formulation, both of them are represented with the $r_e(t)$ variable. The weight of the $r_e(t)$ variable is multiplied by 2 since it accounts for two links at the same time. The $r_e(t)$ variables are found as if the decoding node is the source node of the connections, which are destined to the destination nodes D_i's.

The set of parameters are the same as the CGF algorithm has except that there are two $[0, 1]$ binary parameters derived from the shortest path routing of primary paths. The parameter $y_e(i)$ is eliminated and the context of $m(i, j)$ is altered. The new parameters are

- $x_e(i)$: Equals 1 iff the primary path of connection i traverses over span e, is acquired from the shortest path routing,
- $m(i, j)$: Equals 1 iff the primary path of connection i is link-disjoint to the primary path of connection j, is acquired from the shortest path routing.

The SCP algorithm has the same set of variables as the CGF algorithm.

Equation (10) and inequality (11) are used in the same context as stated in the previous section,

$$\sum_{t=1}^{T} n(i, t) = 1 \qquad \forall i, \qquad (10)$$

$$p_v(t) - p_u(t) \geq \alpha \cdot c_e(t) - (1 - c_e(t)) \qquad \forall e = u \rightarrow v, \forall t. \quad (11)$$

Inequality (12) is similar to the one in the previous section except $m(i, j)$ keeps the track of link-disjointness only between the primary paths of connection i and j,

$$\begin{aligned} n(i, t) + n(j, t) \leq 1 + m(i, j), \\ \forall i, j \in N, i < j, \forall t. \end{aligned} \qquad (12)$$

The link-disjointness between the primary paths and the protection topology in the same coding group is ensured with

$$x_e(i) + x_f(i) + c_e(t) + c_f(t) + r_e(t) + r_f(t) + n(i,t) \leq 2, \quad \forall e, f \in g, \forall g \in S, \forall i \in N, \forall t. \tag{13}$$

Equation (14) ensures that there is only one decoding node on each coding group,

$$\sum_{v \in V} s_v(t) = 1 \qquad \forall t. \tag{14}$$

The protection tree of each coding group is found by treating the decoding node of the coding group as the destination node of the flows originating from the source nodes. The behavior of the protection tree depends on the node. The protection tree must be input to the decoding node using one of the incoming links. It is stated by

$$\sum_{e \in \Gamma_i(v)} c_e(t) \geq \frac{\sum_{i=1}^{N} n(i,t)}{\beta} + s_v(t) - 1 \qquad \forall v \in V, \forall t. \tag{15}$$

If node v is a source node then the protection tree must originate from this node using one of the outgoing links. If node v is an intermediate node then it must link the incoming protection tree over one of the outgoing links to the decoding node. These two statements are combined and ensured with

$$\sum_{e \in \Gamma_o(v)} c_e(t) \geq \frac{\sum_{i=1, s_i = v}^{N} n(i,t)}{\beta} + \frac{\sum_{e \in \Gamma_i(v)} c_e(t)}{\beta} - s_v(t) \tag{16}$$
$$\forall v \in V, \forall t.$$

The feedback tree is characterized as the union of the protection paths of some hypothetical connections. The source node of these hypothetical connections is the decoding node and the destination nodes are the same destination nodes of original connections. There are the same number of hypothetical connections as the original connection demands. The feedback tree is built by

$$\sum_{e \in \Gamma_i(v)} r_e(t) \geq \frac{\sum_{i=1, d_i = v}^{N} n(i,t)}{\beta} + \frac{\sum_{e \in \Gamma_o(v)} r_e(t)}{\beta} - s_v(t) \tag{17}$$
$$\forall v \in V, \forall t.$$

The objective function is

$$\min \sum_{e \in E} \sum_{t=1}^{T} a_e \times (c_e(t) + 2 \times r_e(t)). \tag{18}$$

The variable $r_e(t)$ is multiplied by 2 since it accounts for both the protection tree between the decoding node and the destination nodes and the feedback tree over the same topology, but in the reverse direction. This portion is depicted with two different gray arrows, one straight and one dashed, in Fig. 2(b).

C. Joint Capacity Placement

The third design algorithm is the Joint Capacity Placement (JCP) algorithm. It is the optimal design algorithm for general diversity coding. However, it has higher complexity than the SCP and the CGF algorithms. Therefore, in this paper, this algorithm is not incorporated in the simulations due to time and resource limitations. In this algorithm, the primary paths are jointly calculated along with coding groups and their protection topologies. The parameters $x_e(i)$ and $m(i,j)$ of the SCP algorithm are converted to $[0,1]$ binary variables. The primary paths are found by

$$\sum_{e \in \Gamma_i(v)} x_e(i) - \sum_{e \in \Gamma_o(v)} x_e(i) = \begin{cases} \text{-1} & \text{if } v = s_i, \\ 1 & \text{if } v = d_i, \quad \forall i \in N. \\ 0 & \text{otherwise}, \end{cases} \tag{19}$$

Inequality (20) calculates the variable $m(i,j)$, which is the link-disjointness between the primary paths, by

$$x_e(i) + x_f(i) + x_e(j) + x_f(j) + m(i,j) \leq 2 \quad \forall e, f \in g, \forall g \in S, \forall i,j \in N, i < j. \tag{20}$$

IV. Results

In this section, we report our investigation of the performance of the proposed algorithms compared to SPP [10], p-cycle protection [18], $1+1$ Automatic Protection Switching (APS) [7], Diversity Coding Tree algorithm [12], and CPP [11]. The proposed techniques are potentially more capacity efficient than the Diversity Coding Tree algorithm since the coding is not limited to the connections with the same destination node. However, they incur more design complexity since new variables are defined to handle the decoding operations. Higher design complexity implies bigger optimality gap given the limited computational resources. In these simulations, we test if the enhancement in the coding structure can overcome the additional design complexity in the general diversity coding algorithms.

In order to analyze the performance of the proposed algorithms, we ran simulations on the NSFNET test network using CPLEX 12.2. This network is depicted in Fig. 3, in which the numbers next to the nodes are node indices and the numbers next to the links are the costs (lengths) of using that link. The main performance metrics are the capacity efficiency and the worst-case restoration time (RT). Capacity efficiency is quantized by calculating the total capacity (TC) required to route and protect the traffic. The traffic matrix of the NSFNET network consists of 300 random unit-sized connection demands, which are chosen using a realistic gravity-based model [19]. In this model, the traffic between two nodes is proportional to the multiplication of their population. The design algorithm for SPP is taken from [7, p. 406]. The design algorithm for p-cycle protection is the optimal cycle-exclusion

TABLE I
SIMULATION RESULTS OF NSFNET NETWORK

NSFNET Network, 14 nodes, 21 spans					
Scheme	TC	RT for different X values (ms)			
		0.5ms	1ms	5ms	10ms
Div. Cod. Tree	1641040	0.03			
CPP	1531179	36.96			
Div. Cod. CGF	1557400	0.03			
Div. Cod. SCP	1542337	0.03			
$1+1$ APS	1881880	0.02			
SPP	1264865	82.87	83.37	87.37	92.37
p-cycle	1440435	74.41	74.91	78.91	83.91

based ILP for SCP taken from [18]. $1+1$ APS is calculated using *2*-shortest link-disjoint paths routing algorithm in [7, p. 196].

The variable X represents the optical cross-connect (OXC) configuration time. The restoration time of the general diversity coding schemes can be decreased to sub-ms using synchronization and buffering as in [12].

The simulation results are given in Table I. As expected, the proposed algorithms result in higher restoration speed but lower capacity efficiency than the SPP, the p-cycle protection, and the CPP algorithms. The difference in terms of capacity efficiency is not significant compared to the speed advantage of diversity coding techniques, considering a difference of three orders of magnitude in restoration time. The outcomes are the opposite for the $1+1$ APS. We note that APS will have even more capacity inefficiency in more populated networks. An important observation is the fact that the proposed algorithms have higher capacity efficiency than the Diversity Coding Tree algorithm. This shows the advantage of coding flexibility. Moreover, the diversity coding SCP even further results in higher capacity efficiency than diversity coding CGF since it optimizes both spare capacity allocation and coding group formation without introducing any extra complexity on top of the diversity coding CGF algorithm. As a result, increased coding flexibility and increased design optimality enables achieving higher capacity efficiency.

V. CONCLUSION

In this paper, we have developed three MIP formulations for general diversity coding design algorithms. We know that these algorithms have increasing optimality and complexity. At the same time, they result in very fast restoration as fast as our former fastest algorithm, Diversity Coding Tree. Our results confirm our algorithm design. We were able to run our algorithms on a desktop computing platform although the design process can take days. As the computational power of such platforms increase, or by employing a very fast computing platform, the algorithm will be able to easily generate solutions for larger networks than we employed in our simulations. As a future work, heuristic algorithms can be developed and advanced ILP methods can be used to cope with the design complexity better.

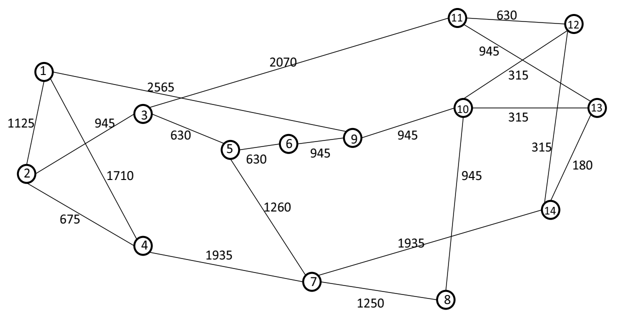

Fig. 3. NSFNET network.

REFERENCES

[1] E. Ayanoglu, C.-L. I, R. D. Gitlin, and J. E. Mazo, "Diversity coding: Using error control for self-healing in communication networks," in *Proc. IEEE INFOCOM '90*, vol. 1, June 1990, pp. 95–104.

[2] ——, "Diversity coding for transparent self-healing and fault-tolerant communication networks," *IEEE Trans. Commun.*, vol. 41, pp. 1677–1686, November 1993.

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

[4] S. Avci, X. Hu, and E. Ayanoglu, "Recovery from link failures in networks with arbitrary topology via diversity coding," in *Proc. IEEE GLOBECOM*, December 2011, pp. 1–6.

[5] ——, "Hitless recovery from link failures in networks with arbitrary topology," in *Proc. of the UCSD Information Theory and Applications Workshop*, February 2011, pp. 1–6.

[6] Y. Xiong and L. G. Mason, "Restoration strategies and spare capacity requirements in self-healing ATM networks," *IEEE/ACM Trans. Netw.*, vol. 7, pp. 98–110, February 1999.

[7] W. D. Grover, *Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking*. Prentice-Hall PTR, 2004.

[8] A. Grue and *et al.*, "Comparative study of fully pre-cross-connected protection architectures for transparent optical networks," in *Proc. IEEE 6th International Workshop on Design and Reliable Communication Networks*, 2007, pp. 1–8.

[9] B. Wu, P.-H. Ho, K. L. Yeung, J. Tapolcai, and H. T. Mouftah, "CFP: cooperative fast protection," *J. Lightwave Technol.*, vol. 28, no. 7, pp. 1102–1113, April 2010.

[10] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, "Survivable WDM mesh networks," *J. Lightwave Technol.*, vol. 21, no. 4, pp. 870–883, April 2003.

[11] S. Avci and E. Ayanoglu, "Coded path protection: Efficient conversion of sharing to coding," in *Proc. IEEE ICC*, June 2012, pp. 1–6.

[12] ——, "Optimal algorithms for near-hitless network restoration via diversity coding," in *Proc. IEEE GLOBECOM to appear*, December 2012, pp. 1–7.

[13] ——, "Design algorithms for fast network restoration via diversity coding," in *Proc. of the UCSD Information Theory and Applications Workshop*, February 2012, pp. 1–7.

[14] ——, "Extended diversity coding: Coding protection and primary paths for network restoration," in *Proc. of the International Symposium on Network Coding*, June 2012, pp. 1–6.

[15] A. E. Kamal, "1+N network protection for mesh networks: Network coding-based protection using *p*-cycles," *IEEE/ACM Trans. Netw.*, vol. 18, no. 1, pp. 67–80, Feb. 2010.

[16] A. E. Kamal and O. Al-Kofahi, "Efficient and agile 1+N protection," *IEEE Trans. Comm.*, vol. 59, no. 1, pp. 169–180, January 2011.

[17] A. E. Kamal, A. Ramamoorthy, L. Long, and S. Li, "Overlay protection against link failures using network coding," *IEEE/ACM Trans. Netw.*, vol. 19, no. 4, pp. 1071–1084, Aug. 2011.

[18] B. Wu, K. L. Yeung, and P.-H. Ho, "ILP formulations for *p*-cycle design without candidate cycle enumeration," *IEEE/ACM Trans. Netw.*, vol. 18, no. 1, pp. 284–295, February 2010.

[19] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale IP traffic matrices from link loads," in *Proc. ACM SIGMETRICS*, June 2003.