All-Optical Network Coding

Eric D. Manley, Jitender S. Deogun, Lisong Xu, and Dennis R. Alexander

***Abstract*—In this paper, we investigate the application of network coding to all-optical networks from both the algorithmic and infrastructural perspectives. We study the effectiveness of using network coding for optical-layer dedicated protection of multicast traffic which provides robustness against link failures in the network. We present a heuristic for solving this problem and compare it with both inefficient optimal methods and non-network coding approaches. Our experiments show that our heuristic provides near optimal performance while significantly outperforming existing approaches for dedicated multicast protection. We also propose architectures for specialized all-optical circuits capable of performing the processing required for network coding and show how these devices can be effectively deployed in an all-optical multicast network.**

I. Introduction

WITH new high-definition video technologies and data-intensive applications, it appears that our increasing need for the large amounts of bandwidth offered by WDM optical networks will not ebb any time soon. Network coding has begun to alter the way we think about communication networks. The disconnect between the theory of network coding and the implementation realities of optical networking are only just beginning to be rectified. In this paper, we propose infrastructure designs to bring all-optical network coding closer to reality and investigate the application of network coding to optical-layer multicast protection. In order to properly motivate this study, we first give a brief overview and background on both network coding and optical multicast.

A. Network Coding

Network coding is well-known for its potential to increase throughput of multicast connections [1]. However, it can also protect against the failure of network components such as nodes or links. The concept employed in this case is called *static network coding*, which we give an example of in Fig. 1. This example shows a four-node network, with two source nodes A and B, and one destination node D. Node A transmits a continuous stream a of bits while node B sends stream b. We want to set up a connection which is robust against the failure of a single link; thus, a single path from a source to the sink is not sufficient. Therefore, both source nodes send their streams to node C, and node C combines bits from both streams using XOR before forwarding to node D. If link (A, D) fails, then D may still recover a through an XOR of the $(a \oplus b)$ stream received from C with stream b received from B. Similarly, D may recover b if link (B, D) fails. Without network coding, dedicated protection is not possible for this network.

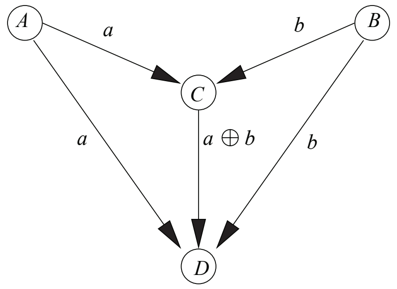


Fig. 1. A static network code robust against a single link failure.

B. Optical Multicast

With the emergence of high-bandwidth point-to-multipoint applications like high definition Internet television, video conferencing, and storage area networks, the need has arisen for optical multicast networks. Thus, optical cross-connects (OXCs), like the one in Fig. 2, supporting multicast at the optical layer have been developed [2], [3]. The key component supporting multicast is a $1 \times F$ power splitter, a passive device which takes the input signal and copies it onto F different waveguides. This fan-out causes the signal power to be split F ways requiring amplification of the output. Usually, full F fan-out is not needed, so only signals which are needed at the output ports need to be amplified. A reconfigurable splitter can be controlled to reduce the splitting which reduces the need for amplification [4]. Other multicast switching technologies include splitter-and-delivery switches (SaD) [5], multicast-only SaD [6], and tap-and-continue [7].

The rest of this paper is organized as follows. In Section II, we discuss some related work. We formulate the problem in Section III, and we investigate the algorithmic aspect of the problem in Section IV. Our infrastructural designs are then presented in Section V, and we conclude in Section VI.

II. Related Work

In this section, we first look at some network coding theory for multicast in both the non-robust and robust cases. We then discuss protection of optical multicast connections and review some research applying network coding to optical networks.

A. Network Coding

Network coding was introduced in 2000 by Ahlswede *et al.* whose main result says that for a network $G = (V, E)$ (a multigraph) with source s and sink set T, the maximum

E. Manley is with the Department of Mathematics and Computer Science, Drake University, Des Moines, IA, 50311 USA e-mail: eric.manley@drake.edu .

J. Deogun and L. Xu are with the Department of Computer Science and Engineering, University of Nebraska at Lincoln, Lincoln, Nebraska 68588-0115.

D. Alexander is with the Department of Electrical Engineering, University of Nebraska at Lincoln, Lincoln, Nebraska 68588-05111.

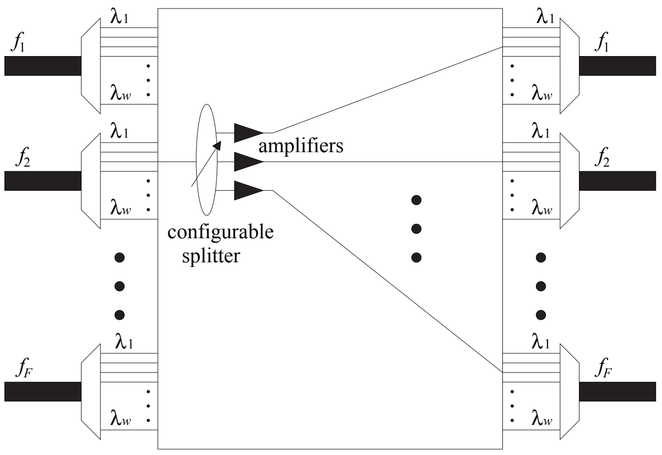


Fig. 2. A multicast-capable OXC.

multicast rate achievable is $h = \min_{t\in T} \text{max\_flow}(G, s, t)$, which means that if the network admits a flow of size h' individually in different networks for the source paired with each sink, then a multicast of rate h' can be achieved to all the sinks simultaneously. This rate is not achievable in traditional store-and-forward network models [1].

With *linear network coding*, each intermediate node transmits linear combinations of the symbols received on incoming channels. Li *et al.* showed that linear network coding is sufficient to achieve the maximum flow rate in the single-source problem [8]. Koetter and Médard gave an algebraic formulation in which many of the foundational theorems result from established theory in algebra [9].

A simple randomized network coding algorithm was proposed by Ho *et al.* in which all coefficients in the linear combination are selected randomly from some finite field $\mathbb{F}$. The probability that all sinks can decode the message is $(1 - |T|/|\mathbb{F}|)^{\nu}$ where ν is the maximum number of edges with flow in a solution [10]. Thus, with large enough symbol sizes, this is a simple, practical, and efficient solution.

Jaggi *et al.* presented both deterministic and randomized polynomial time algorithms for finding network codes when they exist. For a directed, acyclic multigraph with unit capacities, the deterministic algorithm has running time $O(|E| \cdot |T| \cdot h \cdot (h + |T|))$ and requires that that coding operations be done in a field of size at least $|T|$. The randomized algorithm has expected running time $O(|E| \cdot |T| \cdot h^2)$ but requires a field of size at least $2|T|$ [11].

The multiple-source network coding problem is significantly more difficult than the single source problem, so multiple sessions are often supported using disjoint subgraphs of the network. Lun *et al.* investigated finding a subgraph of the available topology which is then given as input to the network coding algorithm [12]. They modeled this problem as a linear program which finds the asymptotic optimal solution in polynomial time [12].

Fragouli and Soljanin proposed a method for identifying equivalence classes of network topologies based on their decomposition into subgraphs which carry the same information in a particular network code [13]. This leads to both simplified decentralized network coding algorithms as well as a means of determining the size of the required finite field of the encoded symbols, both of which are important when applying network coding in the optical domain.

B. Robust Network Coding

Koetter and Médard formulated the network coding problem for robustness against a given collection of link failure patterns [9]. A link failure pattern is a subset of the edge set E, and signifies those edges which may fail simultaneously. A network coding solution is defined as *static* under a collection $\mathcal{F}$ of failure patterns if none of the coding coefficients must change and all messages can still be decoded by all sinks after a failure. They showed that for a single-source multicast session on the network represented by multigraph $G = (V, E)$, there exists a static network code under $\mathcal{F}$ if there exists a network coding solution for the multicast on graph $G_f = (V, E \setminus f)$ for all $f \in \mathcal{F}$. They also showed that this result does not hold for an arbitrary set of connections. For a transmission rate of r (i.e. r symbols to be decoded), Koetter and Médard's algorithm requires a finite field of size at least $r|T||\mathcal{F}|$. A randomized algorithm of Jaggi *et al.* requires a field of size $2|T||\mathcal{F}|$ with expected running time $O(|\mathcal{F}||E|(r^2|T| + r \cdot \min\{\max_{v\in V} \deg(v), |T||\mathcal{F}|\}))$. Another version of the algorithm computes static codes with probability at least $1-\delta$ if the finite field has at least $|E||T||\mathcal{F}|/\delta$ elements [11]. Yeung *et al.* notes that a deterministic static network coding algorithm which is polynomial in terms of $|\mathcal{F}||E|$ can be constructed given a sufficiently large field [14].

C. Protection of Optical Multicast

To make use of optical multicasting technology, the routing and wavelength assignment problem has been extended for multicast [15]. Multicasting brings a new level of complexity as multicast routing requires that a Steiner tree (i.e. a tree spanning the source and sink set of nodes) be computed. However, the Steiner tree problem is itself NP-Complete while path computation is solvable in polynomial time [16].

Optical protection is the concept of pre-computing and provisioning backup bandwidth which is used in case of a failure. For protection of optical multicast, primary light-trees must be protected with backup light-trees. To achieve dedicated protection, redundant trees must be computed. In its most simple formulation, to protect against a single-link failure, the backup tree is link-disjoint from the primary tree. However, structures of communication networks do not necessarily lend themselves well to Steiner-tree packing.

Singhal and Mukherjee formulated this problem for WDM optical multicast and presented multicast-capable OXC architectures for supporting it both all-optically and with optical-electrical-optical (OEO) conversion. They then showed how a primary and backup tree may be able to use the same links in opposite directions (removing the some of the link-disjointness requirements) which may lead to better solutions [17]. Rahman and Ellinas studied dedicated mulitcast protection for optical networks and presented three heuristics for packing two Steiner trees to achieve dedicated protection in the single-link failure model. The best performing of their heuristics was the *Minimum Cost Collapsed Ring (MCCR)* heuristic which attempts to use the robust nature of WDM rings by finding a two-wavelength ring containing the source and destination nodes. This approach also exploits the ability to use the same links in opposite directions [18]. Boworntummarat, Leelarusmee, *et al.* consider dedicated protection with multiple fibers on each physical link [19]. Singhal, Sahasrabuddhe, and Mukherjee proposed an approach to multicast protection called *self-sharing trees* in which backup bandwidth is dedicated for a particular connection but is multiplexed against different link failure patterns on the primary tree [20]. Singhal, Ou, and Mukherjee then extended this idea to *cross-sharing* in which two different multicast sessions share backup bandwidth [21]. This shared protection provides better bandwidth efficiency at the cost of slower recovery times due to switch reconfiguration time and propagation delay.

D. Network Coding in Optical Networks

Much of the research on using network coding for optical networks relies on OEO conversion with electronic buffering and processing at each node. The use of photonic circuits for this purpose is only just beginning to be investigated.

Ahmed Kamal conducted the first research into applying network coding to optical unicast protection [22], [23], [24], [25]. He provides some sufficient conditions for which static network coding can be provided in the multi-source unicast case. He shows that this basic approach provides many benefits over traditional protection schemes. Because static network coding has the benefits of $1+1$ protection but still allows a single unit of backup bandwidth to be multiplexed over N connections, this approach is called $1+N$ protection.

Originally, Kamal presented $1+N$ protection using the structure of p-Cycles [22]. The OEO, buffering, and processing time then impacts the apparent recovery time, so it should be small compared to the detection and retransmission time of the alternative non-coding method. Kamal later extended this approach to handle multiple failures [23]. He also gave a hybrid $1+N$ scheme employing p-Cyles and GMPLS standard Label Switched Paths (LSP) for protecting the on-cycle links and a modified LSP for protecting straddling links [24]. Kamal also presented a generalized version of the approach, which does not use p-Cycles [25]. In this scheme, a set of primary connections is protected by a backup tree over which all the connections have coded their packets. If the connections do not have a common sink, then the sinks each relay enough decoding information over a set of *collector* and *delivery* links in order to recover all data upon a failure. This Kamal and Al-Kofahi extended this idea for protection of bidirectional connections with no additional bandwidth needed over non-network-coding shared protection schemes [26]. Al-Kofahi and Kamal also characterized networks admitting static network codes for multiple-sources with a common sink and applied it to protection of wireless flows [27]. Kamal and Ramamoorthy proposed another generalization for multiple-link failures via implementation in an overlay layer which required fewer resources and simpler synchronization [28].

Menendez and Gannet propose using photonic XOR devices for network coding and show its operate with cross-session coding of two multicast sessions with a shared sink set in a simple network topology. In addition to link failure, they also take into account failure of the XOR device [29].

Our previous work on applying network coding to all-optical networks considered the coarse-grained version of Lun *et al.*'s [12] subgraph problem for all-optical multicast [30]. Kim *et al.* considered the coarse-grained static network coding problem for optical networks [31]. They looked at both minimizing link cost and coding costs in separate stages as well as a combined evolutionary approach. Their experiments showed that in most cases, the optimal network coding subgraph requires no coding at all which corresponds to a similar observation made by Li, Li, and Lau for the non-robust case that there is little benefit to network coding when compared to optimal multicast trees [32]. However, since computing both optimally is NP-Complete in general, the real power of network coding comes from its potential to improve heuristic performance.

III. Problem Formulation

In this Section, we formulate both the infrastructure design and algorithmic problems which we address in later sections.

A. All-Optical Network Coding Infrastructure Requirements

In order for network coding to be implemented in an all-optical network, network infrastructures must be designed which allow for network coding operations.For linear network codes, this requires the computation of a linear combination over the input symbols for each outgoing channel. This is a relatively straightforward operation in electronic networks which can buffer packets and perform the computations with readily available ALU operations. To be realized directly on an optical network, the signal would have to be terminated, converted to electronics, and buffered before computation and retransmission at each node. Since this conversion and "stopping" of the bit stream is undesirable for optical WDM mesh networks, infrastructure must be built which allows the network to perform these operations seamlessly in the optical domain. This infrastructure requires devices for various all-optical operations. These devices must be designed for computing scalar multiplication with a fixed (i.e. the same for every packet in a bit stream, although it may vary for different connections) coefficient as well as addition of variable symbols encoded on the optical medium. We will now discuss the enabling technologies on which such devices should be based.

1) Optical Switching Technologies: We assume that the technology which enables multicast-capable OXCs is available for network coding devices. Some of these devices include passive splitters [33], combiners [34], amplifiers [35], wavelength converters [36], and small (e.g. 2×2, 2×1, 1×2) all-optical switches [37].

2) All-Optical Buffers: Although we do not allow for electronic buffering of the data, some buffering may occur in the optical domain. We do not rely on random-access technology, but merely tunable delay of the optical signal. All-optical buffers have been proposed for use in certain applications such as Optical Packet Switching (OPS). These buffers are typically built using a series of fiber delay lines onto which the optical signal can be switched [38], but more sophisticated technology based on *slow-light* effects in optical microcavity resonators have also been investigated. This technology uses an external light source to controllably slow down the light by altering its dispersion characteristics [39], [40].

3) All-Optical Logic Gates: Architectures for optical computing have been proposed which take advantage of the fact that 2×2 switches are sufficient for realizing all possible logic gates (assuming the existence of constants 0 and 1 as well as fan-out which can be implemented with a passive splitter) [41]. Furthermore, researchers have investigated designs for logic gates suitable for optical processing. Several all-optical XOR gates have been demonstrated using semiconductor optical amplifiers (SOAs) as interferometers. For instance, an all-optical XOR gate operating at 40 GHz was demonstrated by Theophilopoulos *et al.* in an SOA-based ultrafast nonlinear interferometer gate [42]. Webb *et al.* demonstrated a 40 Gbit/s all-optical XOR with an SOA based Mach-Zehnder interferometer which allows pulse shaping [43]. Wang *et al.* also demonstrated an SOA-based Mach-Zehnder implementation, but addressed the problem of carrier lifetime limitations imposed by the SOA [44]. Their demonstration operated at 20 and 40 Gbit/s, but they claim that it is suitable for even over 100 Gbit/s. For an overview of how SOAs can be used for all-optical logic gates, see [45]. For a survey of all-optical XOR technologies, see [46].

B. Algorithmic Problems

Once network coding operations are supported at the optical layer, it is important that we determine how these services can be used by the coding algorithms which replace the routing algorithms in traditional optical networks.

As discussed in Section II-A, Lun *et al.*'s formulation of the network coding subgraph problem as a linear program is asymptotically optimal in polynomial-time [12]. This method is practical for electronic packet-based networks which can split a message into arbitrary fractions across different communication channels. However, with a coarse bandwidth granularity, this method will not necessarily produce usable solutions. In optical networks, the bandwidth granularity is usually extremely coarse, often with a minimum subdivision on the order of tens of Gbps. Thus, for our purposes, it is more appropriate to model the problem as a flow problem on a multigraph in which each edge represents the minimum indivisible unit of supported bandwidth in the network (e.g. the capacity of one wavelength).

Since Koetter and Médard proved that a static network code exists for a single-source multicast under a set of link failure patterns if and only if network codes exist after individually removing each failure pattern [9], we have a condition for the existence of a static network code which is robust against any set of κ simultaneous link failures. Note that each wavelength channel is represented by an individual edge in a multigraph, and we define a *link* as the set of multiple edges which share a common pair of endpoints but not necessarily oriented in the same direction. We assume that the failure of a link severs all channels on all fibers between a pair of nodes. If it is desired that we protect against specific fibers within a multi-fiber network, the model can be adapted. Our model also assumes that each node has identical capabilities and has the ability to perform all needed network coding operations. We also assume full wavelength conversion capability, and we do not limit the splitting ratio.

Thus, for a rate r transmission, we need to guarantee that there exists r edge disjoint paths between s and t for each $t \in T$ after the failure of κ links. This is true for every cut in the graph, which means that removing any κ links from a cut, the remaining links must have at least r edges spread over them. However, when dealing with only single unit multicasts, this degenerates to a simpler link-disjointness condition.

Problem 1: Minimum Cost Static Integral Coded Flow
Instance: A directed weighted multigraph $G = (V, E)$, a source vertex $s \in V$, a set of sink vertices $T \subseteq V$ with $s \notin T$, and a positive integer κ.
Question: Find a minimum cost subgraph $G' = (V', E')$ of G with $T \cup \{s\} \subseteq V'$ such that for each $t \in T$, G' contains $\kappa + 1$ link-disjoint paths between s and t.

This problem is NP-Complete and hard to approximate within a factor of $\log(|T|)$ for any fixed κ via a reduction from the directed Steiner tree problem [47].

IV. Protected Multicast

Protected multicast is difficult to provide. For instance, the MCCR heuristic proposed by Rahman and Ellinas had a blocking probability of over 30% for even a size 3 multicast group on NSFNET with 64 wavelengths and a load of 100 Erlang, and blocking probability tended to increase greatly with increase in load or group size [18]. The problem is that disjoint multicast trees do not always exist and may be difficult to find when they do. Thus, many approaches often do not find usable solutions, and when they do, the solutions tend to take up a lot of resources. Since Kamal *et al.* has had success with $1 + N$ protection for unicast [22], it is natural to ask if network coding can provide better protection for multicast sessions. We now present our heuristic and give results from our simulations.

A. RCM Heuristic

Our approach generalizes strategies for solving the Steiner tree problem. The classic shortest path heuristic (SPH) [48], [49] for finding Steiner trees in undirected networks builds the tree iteratively by first finding the shortest path among all pairs of nodes in the terminal set. This path is added to the tree. Then, the shortest distance between each remaining terminal and the partially computed tree is found, and the path of shortest distance is added to the tree. This continues until all terminals have been connected to the tree. This same idea can be applied to directed networks although it ceases to be a 2-approximation algorithm. We can also extend the approach for $(\kappa+1)$-link-connectivity rather than the simple connectedness required by the Steiner tree problem.

In RCM, detailed in Algorithm 1, the subgraph is built incrementally starting with a Steiner 1-link-connected subgraph which then increments to a Steiner 2-link-connected graph and so on. When incrementing the subgraph for Steiner i-link connectivity for a particular sink node, we cannot use any of the $i-1$ paths previously computed because this would destroy the link-disjointness of the computed paths to that sink, so these should be temporarily deleted from the graph. Any other edge which has been allocated for another path may be used for the current sink without adding cost to the current solution, so these edges should temporarily be given zero cost. This can be implemented by maintaining a list of all paths computed from s to t for all $t \in T$ in a $|T| \times r$ path matrix. Such a matrix is initialized on line 2 and updated through each iteration of the main loop beginning on line 3. For a given iteration, we create a new auxiliary graph (line 11) and then remove all edges from the sink's entries in the path matrix (lines 12-16). We then set the weight of all other edges in the path matrix to 0 in lines 17-19. We compute a path from the source to a new sink in line 20 to get the least-cost increase to the total solution for that sink. Lines 11-20 are repeated (using the loop beginning on line 9) with a new copy of the graph (set on line 10) for each sink node. Lines 21-25 keep track of the cost, path, and sink node (using variables initialized in lines 6-8) corresponding to the lowest cost augmentation of the subgraph among all possible sinks, a change which is made permanent after exiting this loop on line 27. The sink node is then removed from the sink set (for this iteration of the line 3 for-loop), and we repeat using the loop beginning on line 5 until all sinks have a new path added to the subgraph. The algorithm terminates and returns the subgraph (line 31) when we have reached the desired level of Steiner connectivity.

RCM has time complexity $O((\kappa+1)|T|^2(|V|+|E|))$. To see this, note that each edge in the current portion of the subgraph H_0 is examined exactly once between lines 12 and 19 (either for deletion or setting of zero weight in G_0) because each of $p_{v,1}, p_{v,2}, \ldots, p_{v,i-1}$ are link disjoint. It is also evident that each of lines 10, 11, 20 (using some linear-time shortest path algorithm), and 21-25 take $O(|V|+|E|)$. This whole block from line 10 to 25 must be executed $(\kappa+1)|T|^2$ times. This dominates the running time, so the time complexity of the algorithm is $O((\kappa+1)|T|^2(|V|+|E|))$. Since a static network coding algorithm must then be applied, the overall complexity of this approach is impacted by whichever random or deterministic algorithm is selected.

When $\kappa = 1$, if the source is 2-link-connected to each sink and the shortest-path algorithm used in line 20 is altered to instead find the shortest path which does not block all other $s-v$ paths (e.g. run Suurballe's algorithm [50] for finding the shortest path pair and use only the shorter of the two paths), then this algorithm is guaranteed to find a solution.

Algorithm 1 RCM: Robust Coded Multicast

```
Input: directed multigraph $G$, source $s$, sink set $T$, $\kappa$
    simultaneous link failures to protect against
Returns: subgraph $H$ containing $\kappa+1$ disjoint paths
    between $s$ and $v$ for each $v \in T$
 1: Initialize subgraph $H \Leftarrow (V, \emptyset)$.
 2: Create an initially empty $|T| \times (\kappa+1)$ path matrix $p$
    such that $p_{v,i}$ will store the $i^{\text{th}}$ edge disjoint path from
    $s$ to $v$.
 3: for $i$ from 1 to $\kappa+1$ do
 4:   Set $T_0 \Leftarrow T$.
 5:   while $T_0 \neq \emptyset$ do
 6:     Initialize node $v_{best} \Leftarrow$ null.
 7:     Initialize path $p_{best} \Leftarrow$ null.
 8:     Initialize cost $c_{best} \Leftarrow \infty$.
 9:     for $v \in T_0$ do
10:       Set $G_0 \Leftarrow G$
11:       Set $H_0 \Leftarrow H$
12:       for $j$ from 1 to $i-1$ do
13:         for each edge $(x, y)$ on $p_{v,j}$ do
14:           delete all edges with endpoints $x$ and $y$
              from both $G_0$ and $H_0$
15:         end for
16:       end for
17:       for each remaining edge $e \in H_0$ do
18:         Set the weight of $e$ in $G_0$ to 0.
19:       end for
20:       Find $p_{v,i}$, the shortest $s$-$v$ path in $G_0$.
21:       if cost($p_{v,i}$) $< c_{best}$ then
22:         Set $c_{best} \Leftarrow$ cost($p_{v,i}$)
23:         Set $p_{best} \Leftarrow p_{v,i}$
24:         Set $v_{best} \Leftarrow v$
25:       end if
26:     end for
27:     Set $H \Leftarrow H \cup p_{best}$.
28:     Set $T_0 \Leftarrow T_0 - v_{best}$.
29:   end while
30: end for
31: return $H$
```

B. Simulation

We ran simulations to experimentally evaluate the usefulness of our approach. We next describe the algorithms we used for comparison, our simulation setup, the simulation topologies, our performance metrics, and experimental results.

1) Comparison Algorithms: In order to determine how well the network coding solution performed when compared with non-coding solutions, we implemented two additional non-network coding heuristics. The first is a naive algorithm for finding two link disjoint Steiner trees based on a directed version of SPH. We chose an SPH-based algorithm because our coding algorithm is also based on SPH, and this allows us to isolate the advantage that network coding provides over the base heuristic. We have also chosen another heuristic designed specifically for dedicated protection of multicast which is purported to have good results [18]. We also compare with the optimal solution of Problem 1 computed from an integer linear program (ILP).

a) A Naive Algorithm: We implemented a naive algorithm which attempts to find two link-disjoint directed Steiner trees. This algorithm simply computes one Steiner tree, removes all links containing edges on that tree, and then computes another Steiner tree. This repeats until the desired number of link-disjoint Steiner trees have been computed.

b) MCCR Algorithm: Rahman and Ellinas [18] proposed a series of algorithms for dedicated multicast protection. We implemented Rahman and Ellinas' MCCR algorithm because it was reported to have the lowest blocking probability. This approach uses the inherent robustness of the ring structure to construct a protected multicast. First, a ring is found, and two paths are set up in opposite directions on different wavelengths which each terminate at the last node before the source.

c) Static Network Coding ILP: The optimal static network code for protecting a single-unit transmission against a single link failure can be formulated as an ILP. The network is represented by multigraph $G = (V, E)$ with source s and sink set T. We let $x_{(i,j)_k}$ be a variable which takes on the value 1 or 0 [constraint (5)] indicating whether edge $(i,j)_k$ is in the subgraph. For each edge, we have an auxiliary variable $x^t_{(i,j)_k}$ for each sink $t \in T$ indicating whether that edge is allocated on a path from s to t. Constraint (2) forces the source to have a net outgoing flow of 2 and the sink to have a net incoming flow of 2. Constraint (2) forces all other nodes to have equal incoming and outgoing flow. Thus, the variables $x^t_{(i,j)_k}$ will define a path from s to t for each $t \in T$. Constraint (3) forces $x_{(i,j)_k}$ to be 1 if any of the corresponding $x^t_{(i,j)_k}$ are 1. Therefore, the subgraph defined by the edges for which $x_{(i,j)_k} = 1$ will contain 2 paths from s to t for all $t \in T$. Constraint (6) ensures link disjointness of these 2 paths. This constraint requires that the sum of the allocated edges for a particular sink on a particular link be at most 1. Since each of the terms in the sum is an integer 0 or 1, at most one of them will be 1. So, each sink has at most one edge allocated to it on any given link which forces link disjointness of the two paths. We distinguish different edges between a pair of nodes using a subscript. That is, $(i,j)_1, (i,j)_2, \ldots, (i,j)_\sigma$ are the σ edges on link (i,j). Our formulation assumes that each of these may have a different weight (i.e. edge cost) denoted $w_{(i,j)_k}$ for edge $(i,j)_k$, but for our purposes, all multiple edges on the same link will have the same weight.

$$\text{minimize} \sum_{(i,j)_k \in E} w_{(i,j)_k} x_{(i,j)_k} \tag{1}$$

subject to

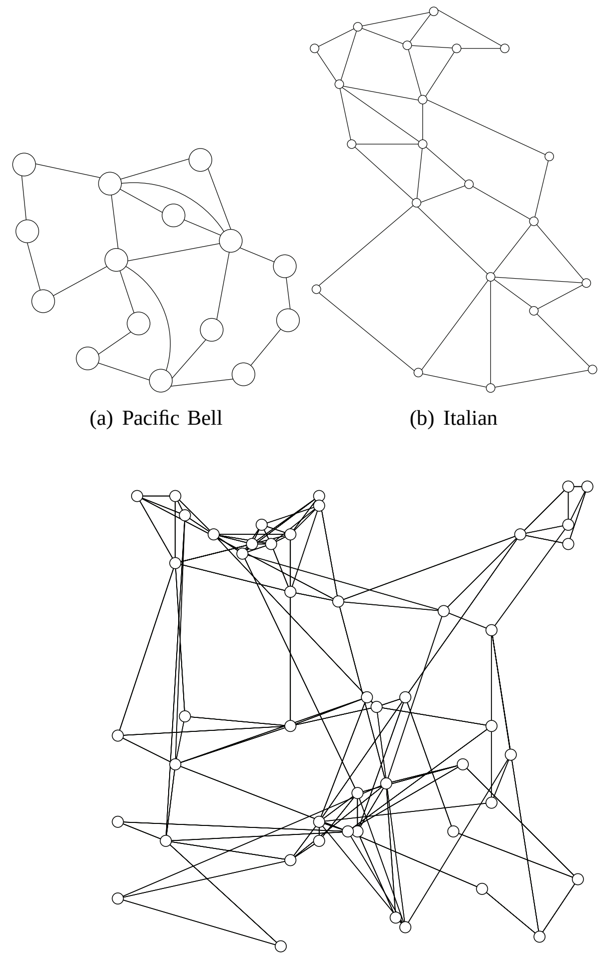
$$\textstyle\sum_{(i,j)_k \in E} x^t_{(i,j)_k} - \sum_{(j,i)_k \in E} x^t_{(j,i)_k} = \begin{cases} 2 & \text{if } i = s \\ -2 & \text{if } i = t \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in V, t \in T \tag{2}$$

$$x_{(i,j)_k} \geq x^t_{(i,j)_k} \quad \forall (i,j)_k \in E, t \in T \tag{3}$$

$$1 \geq x^t_{(i,j)_k} \geq 0 \quad \forall (i,j)_k \in E, t \in T \tag{4}$$

$$1 \geq x_{(i,j)_k} \geq 0 \quad \forall (i,j)_k \in E \tag{5}$$

$$\sum_{(i,j)_k \in E} x^t_{(i,j)_k} \leq 1 \quad \forall i,j \in V \text{s.t. } (i,j)_1 \in E, t \in T \tag{6}$$

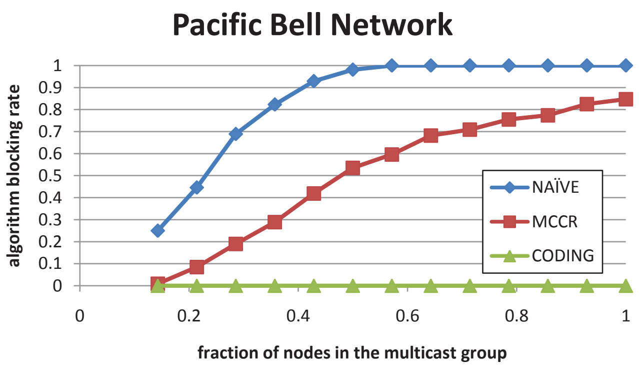

(a) Pacific Bell (b) Italian

(c) Random 50 Node Network A

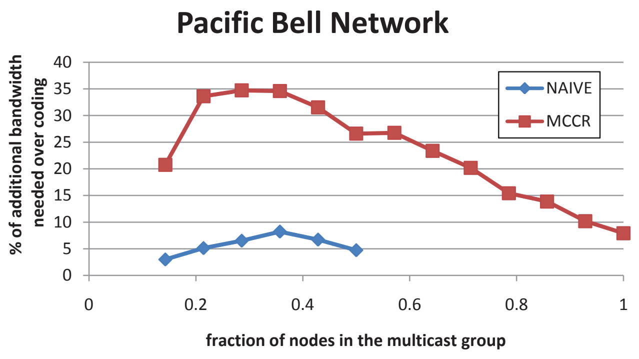
Fig. 3. Topologies used in the simulations.

2) Simulation Setup: We used an incremental traffic model and assumed the existence of enough wavelengths to support all connections. For various numbers of receiving nodes, we generated single-unit bandwidth multicast requests by selecting the source and receiving nodes uniformly at random. These requests are to be protected against a single-link failure. Our first set of experiments compare our heuristic with both the MCCR and naive heuristics. For these experiments, each data point presented is the average over 1000 different multicast sessions. We also compare the performance of our heuristic to the optimal solution discovered by solving the above ILP. In this set of experiments, each data point is the average over 1000 different sessions except for the random 50 node network which we limited to 50 due to long running times.

3) Simulation Topologies: We present results on the three topologies shown in Fig. 3: the 15-node Pacific Bell network, the 21-node Italian network [51], and a 50-node network randomly generated using the rectangular grid method [52]. The Pacific Bell network and the Italian network were chosen to study performance on real-world topologies, while the random network was selected to give a wider range of network sizes. All links are assumed to have unit weight.

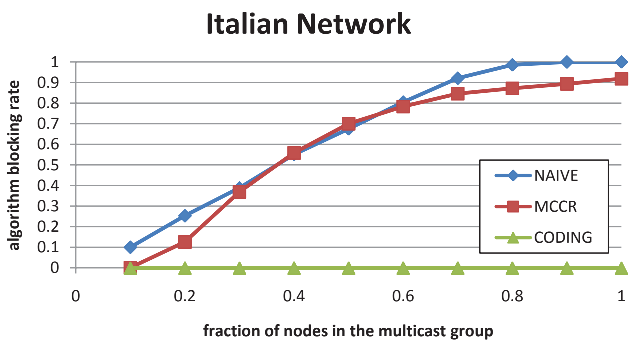


(a)

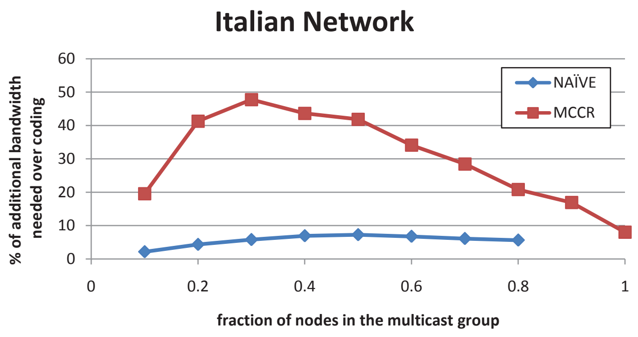


(b)

Fig. 4. For the Pacific Bell network; (a) the algorithm blocking rate for each of the naive heuristic, MCCR heuristic, and our RCM coding heuristic, (b) the amount of additional bandwidth needed by the naive heuristic (resp. MCCR heuristic) for those sessions in which both the naive and coding heuristics (resp. MCCR and coding heuristics) found a valid solution.



(a)



(b)

Fig. 5. For the Italian network; (a) the algorithm blocking rate for each of the naive heuristic, MCCR heuristic, and our RCM coding heuristic, (b) the amount of additional bandwidth needed by the naive heuristic (resp. MCCR heuristic) for those sessions in which both the naive and coding heuristics (resp. MCCR and coding heuristics) found a valid solution.

4) Performance Metrics: The blocking rate is defined as the number of requested sessions which cannot be fulfilled over the total number of requested sessions. Since we assume that all links have enough available bandwidth to support all connections, a connection cannot be blocked because some links are saturated from inefficient routing of previous sessions. Thus, all blocking is due to the fact that the topology does not admit a solution (a case in which the network coding approach has an advantage) or because a solution exists and the algorithm fails to find it. The blocking rates of the three algorithms are shown over varying multicast group sizes in Figs. 4(a), 5(a), and 6(a).

After comparing the blocking rates, we consider the total amount of bandwidth used which is measured as the sum of all individual link bandwidths. We compare each of the naive and MCCR algorithms independently against the network coding approach. Since a different set of sessions are blocked by each algorithm, we only look at the subset of connections in which both the naive algorithm and the coding heuristic (respectively MCCR and the coding heuristic) found a solution.Each data point in those plots [Figs. 4(b), 5(b), and 6(b)] is the percent of additional bandwidth needed when using the traditional protected multicast approach versus the network coding approach. That is, we plot

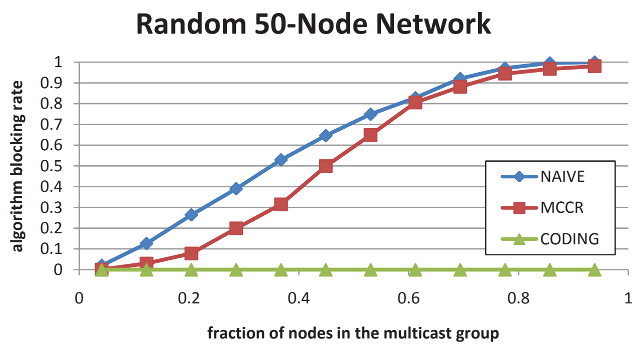
$$\frac{\beta_{\mathrm{nc}} - \beta_{\mathrm{c}}}{\beta_{\mathrm{c}}} \times 100,$$

where β_{nc} is the bandwidth used with non-coding approach and β_{c} is bandwidth used with coding approach.

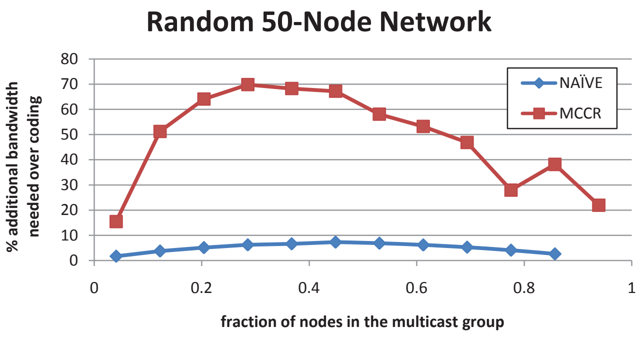
For comparing our heuristic with the optimal method, we simply do a direct comparison of the total amount of bandwidth used by each solution. We also compare the running times of each method in our simulation environment in which the ILP is solved using LpSolve version 5.5 with default settings. All simulation runs were completed on a 2 GHz Intel Core Duo MacBook Pro with 1 GB of 667 MHz DDR2 SDRAM. We provide the timing results because they may have some limited value although we cannot guarantee that the algorithms are implemented in the most efficient way possible. We did not consider any of the time it took to read in the network parameters, generate the multicast groups, or set up the data structures for use by the algorithms.

5) Experimental Results: We now discuss the results from the simulation for both the heuristic and optimal comparisons.

a) Comparison with Dedicated Protected Multicast Routing: In the Pacific Bell network, MCCR blocked significantly fewer sessions than the naive algorithm, although both MCCR and the naive algorithm steadily increased blocking as the size of the groups grew larger. However, the network coding approach outperformed both algorithms, not blocking any of

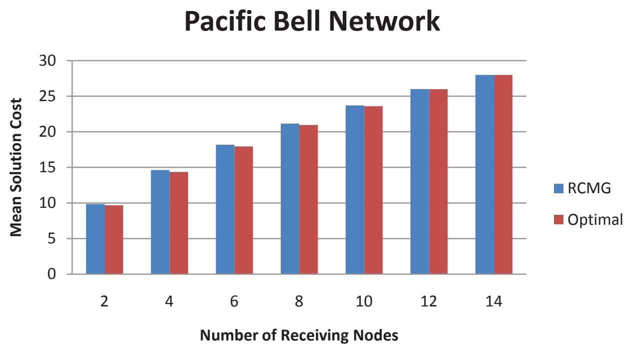


(a)

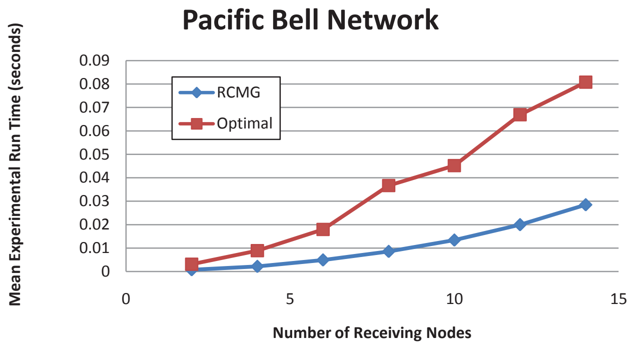


(b)

Fig. 6. For the random 50-node network; (a) the algorithm blocking rate for each of the naive heuristic, MCCR heuristic, and our RCM coding heuristic, (b) the amount of additional bandwidth needed by the naive heuristic (resp. MCCR heuristic) for those sessions in which both the naive and coding heuristics (resp. MCCR and coding heuristics) found a valid solution.



(a)



(b)

Fig. 7. For the Pacific Bell Network, (a) the mean cost of the RCM heuristic solution and (b) the mean experimental running time of the RCM heuristic solution compared with the mean optimal static network coding solution over 1000 different multicast sessions for each group size.

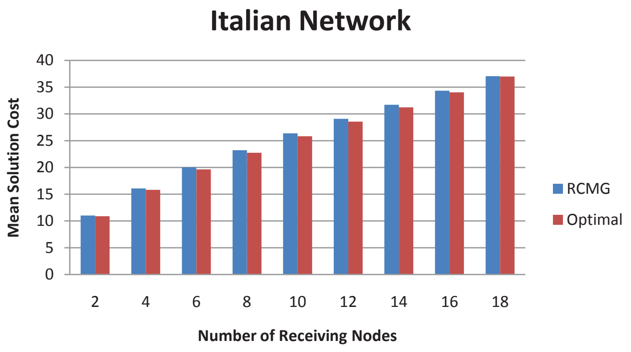
the sessions. For those sessions in which the naive algorithm did not block, it used up to 10% more bandwidth than the coding algorithm [Fig. 4(b)]. Even though the MCCR algorithm blocked less often than the naive algorithm, it used relatively more bandwidth compared with the coding approach.

For the two larger networks, MCCR outperformed the naive algorithm, but to a lesser extent than with the Pacific Bell network. The blocking rate still approached 1.0 with the increase in group size, and the trend in additional bandwidth used continued [Figs. 5(b) and 6(b)] with the naive algorithm using up to 10% more bandwidth. However, MCCR used as much as 50% more bandwidth in the Italian network and as much as 70% more in the random 50-node network.

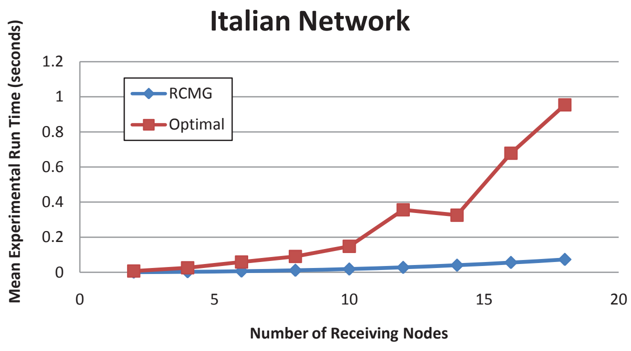
Given these results, it appears that network coding provides a very good solution for robust multicast in optical networks as it has significantly lower blocking and uses less bandwidth than existing approaches. The reason why network coding provides such a drastic improvement is that without network coding, the network must be able to pack in two link-disjoint Steiner trees. This is a global connectivity requirement over all the nodes in the group, and the network may not always have high enough connectivity to guarantee a solution exists, and it is not easy to find such solutions when they do exist. As the multicast group size grows, a single Steiner tree takes up more and more of the network which leaves fewer links to be used by a potential backup tree, so it is not surprising that the non-coding algorithms have high blocking rates as the group size increases. On the other hand, the requirement for network coded protection only requires a certain level of local connectivity (i.e. 2-link-connectivity when $\kappa = 1$) between pairs of nodes. A network which is strongly 2-link-connected will be guaranteed to have a network coding based solution, but it will not necessarily have a Steiner packing solution.

The concave shape of each of the plots in Figs. 4(b), 5(b), and 6(b) is likely due to the fact that as the size of the multicast group grows, the naive and MCCR algorithms each reach a point where they are blocking a significantly large percentage of the connections. Thus, the sample size for our comparison is smaller and represents the easy connections where potential improvement due to network coding is diminished. It has also been shown that (non-static) network coding provides no advantage in throughput in the optimal case for unicast and broadcast connections in undirected networks [53]. It could be that some of this behavior is transferring to this related problem which may partly explain the greater benefit of network coding as the multicast group looks less like either unicast or broadcast.

b) Comparison with Optimal: The results of the comparison between RCM and the optimal solutions are shown in Figs. 7 through 9. RCM achieved near optimal results. On average, RCM was only 0.9%, 1.5%, and 5.0% more costly

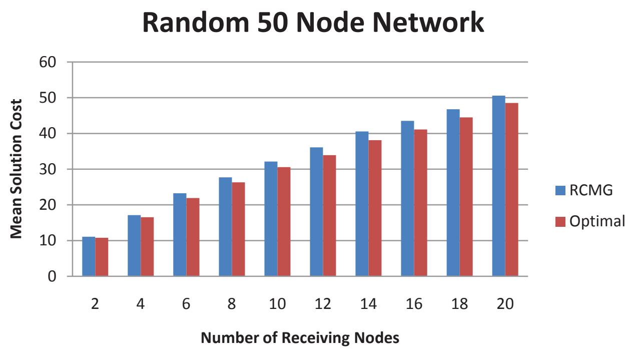


(a)

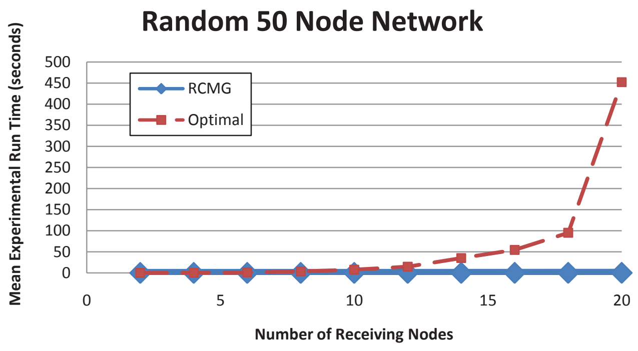


(b)

Fig. 8. For the Italian Network, (a) the mean cost of the RCM heuristic solution and (b) the mean experimental running time of the RCM heuristic solution compared with the mean optimal static network coding solution over 1000 different multicast sessions for each group size.



(a)



(b)

Fig. 9. For the random 50 node network, (a) the mean cost of the RCM heuristic solution and (b) the mean experimental running time of the RCM heuristic solution compared with the mean optimal static network coding solution over 50 different multicast sessions for each group size.

than the optimal solutions in the Pacific Bell, Italian, and random 50 node networks respectively. The worst case for a particular sink set size that we observed in our simulations the heuristic was only 1.7%, 2.1%, and 6.4% more costly.

This slight deviation from the optimal solution comes with a significant savings in computation time. For the Pacific Bell network, the optimal method took between 2.8 and 4.3 times as long as our heuristic on average. For the Italian network it took between 7.3 and 13 times as long. With the larger 50 node network, the gap between computation times increased as the number of sinks increased. With 2 sinks, it took more than 10 times as long on average (0.062 seconds versus 0.0059 seconds). With 20 sinks, RCM terminated in 0.25 seconds on average compared with 452 seconds for the optimal methods, more than 1800 times as long.

We initially chose SPH as the basis for our algorithm because it generalizes naturally for the directed, $(\kappa+1)$-link-connected case. It would be interesting to investigate how better performing Steiner tree heuristics may generalize for this problem, but these comparisons seem to indicate that the room for improvement is somewhat limited.

V. All-Optical Network Coding Infrastructure

In this section, we address the problem of implementing network coding in all-optical networks along with a discussion on OXCs and device placement.

A. Device Deployment

Before we get into the details of computing linear combinations, we look at how such devices could be deployed to offer network coding services at multicast-capable OXCs for both full and limited capability. A device that computes the scalar multiplication part of the linear combination will be referred to as a Scalar Multiplication Unit (SMU) while the addition is accomplished using all-optical XOR gates.

1) Full Network Coding Capability: We first look at devices providing full network coding capability. In general, network coding operations allow for each outgoing channel to be a linear combination over *all* incoming channels. For a $W^{\lambda}(F\times F)$ switch, a switch on F fibers and W wavelengths (see Fig. 10), this means that each of the WF outgoing channels transmit *a sum* of WF different incoming symbols each multiplied by a scalar coefficient.

In this architecture, each of the F fibers enter demultiplexors and each wavelength is split into WF copies using a (re)configurable splitter. After reamplification (not pictured), each signal is buffered with an all-optical buffer which acts as a controlled delay necessary to line up symbols entering on different channels so that when they enter the XOR addition unit, the corresponding bits pass through the device at the same time. The switch will need the ability to time the difference in propagation delay for incoming signals so that

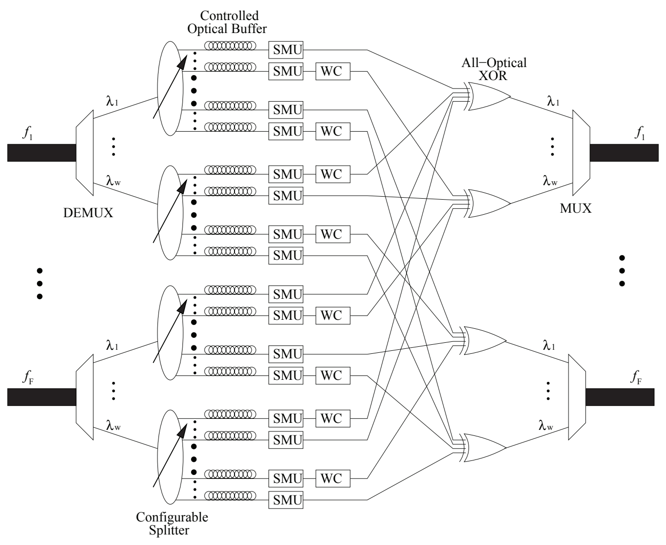


Fig. 10. Multicast-capable $W^{\lambda}(F \times F)$ switch with full network coding capability. Each Wavelength Converter (WC) is a fixed wavelength converter for converting from one specific wavelength to another specific wavelength. Lighter ellipses indicate a repetition of W copies of the pattern while the heavier ellipses indicate an F-copy repetition. Amplifiers which may be needed after power splitting the signal are not shown.

the proper delay can be set on the buffers. Then, the signal enters an SMU where each of the WF copies of the input symbol may be multiplied by a different coefficient and is then sent as one term in the sum of a different output. Since only one wavelength per fiber shares the wavelength with each input, the remaining $W-1$ wavelengths on each fiber utilize a fixed wavelength converter (WC). This fixed wavelength converter will map the input wavelength to the appropriate output channel wavelength before being input to the corresponding XOR adding unit. This design makes the assumption that the SMU and XOR devices can operate at any wavelength and that all inputs to the XOR must be on the same wavelength. Under a different set of assumptions, more or fewer wavelength converters may be necessary. The XOR adding unit has WF inputs (i.e. a copy of each input symbol after undergoing scalar multiplication and possible wavelength conversion) over which it computes the sum. The output of the XOR adding unit is then the linear combination over all input channels. Each of the wavelengths are then multiplexed together to constitute the W channels on each of the F fibers.

Note that for functional correctness, we need not use configurable splitters. Passive splitters are sufficient since the SMU can be configured to multiply the symbol by zero. However, if a significant number of the WF copies of each signal will be zeroed out, we will save on reamplification power as well as the power it takes to operate the buffer, SMU, and WC for that signal by using configurable splitters. Furthermore, a connection not needing any network coding at a given node will have a coefficient of 1 for each output port it is being switched to while all other inputs to the XORs on those ports will have a zeroed out coefficient.

While this design allows for fully functional network coding, its main drawback is its complexity. Each of the SMUs has nontrivial complexity itself, and this design requires $(WF)^2$ of

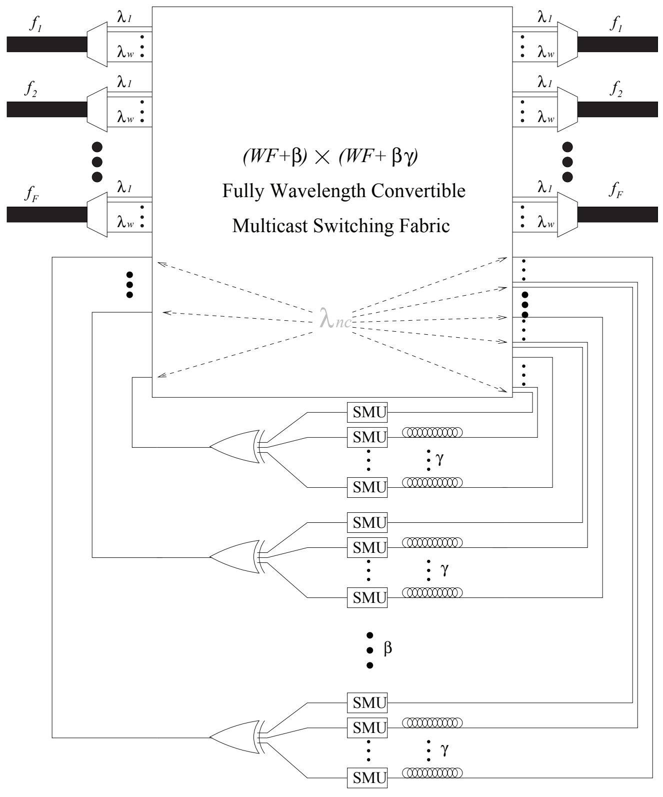


Fig. 11. A multicast-capable $W^{\lambda}(F \times F)$ switch with limited network coding capability. We have not shown the details of the switching fabric which may be any fully wavelength convertible multicast switch. The switching ports leading to and from the network coding circuitry will all be converted to a common wavelength λ_{nc}.

them in addition to the $(WF)^2$ buffers. Each of the WF XORs has WF inputs, which means we need a total $(WF)^2 - WF$ 2-input XORs just for the addition part of the linear combination.

2) Flexible Limited Network Coding Capability: We now give a more flexible version of the OXC architecture allowing limited network coding capability in a share-per-node fashion. This design is shown in Fig. 11 for a $W^{\lambda}(F \times F)$ OXC. The design allows for the use of β simultaneous coding operations to be computed at that node which are each a linear combination of γ terms. The symbol with the highest propagation delay can be switched to the first port of each coding unit and thus need not be delayed by the optical buffer. The switching is all done by a $(WF+\beta) \times (WF+\beta\gamma)$ wavelength convertible switching fabric. The upper portion of the switch acts as a normal wavelength convertible multicast-capable switch which may be used for normal multicast routing in connections not requiring any network coding at this particular node. When network coding is needed, the connections are switched to the lower portion of the switch. In order to allow for simplified computation, all of the network coding outputs are converted to a common wavelength λ_{nc} which is suitable for coding. After being input back into the switch, it may be converted to any needed output wavelength.

This design uses $\beta\gamma$ SMUs, $\beta\gamma - \beta$ optical buffers, and β XOR adders each with γ inputs requiring $\beta\gamma - \beta$ 2-input XORs. The downside is that the switch is larger than the one on which the full network coding capability design is superimposed. This can be mitigated if the switch is designed not to allow multiple paths through the coding circuitry shown on the lower portion of the switch. This would also have the added benefit of preventing excessive signal degradation from spontaneous emissions generated on multiple paths through the coding circuits. In this case, γ should be chosen so that it is large enough to support the number of different channels that may need to be coded together at a given node for a single connection. Then β only needs to be as large as the total number of connections requiring network coding operations at a given node. The comparison of these two approaches is summarized in Table I. This is a significant improvement in complexity for $\beta\gamma << (WF)^2$.

B. All-Optical Coding Circuits

In this section, we discuss how the devices for computing the linear combinations could be built. We first give a brief description of how the symbols are represented. We then present the design of the 2×1 all-optical Linear Combination Unit. We extend this design to the SMU and discuss an alternative design approach that could be used for normalization.

1) Symbol Representation: Because we would like to represent each symbol with a fixed number of bits, all arithmetic operations will occur in GF(2^m), the finite field with 2^m elements. Symbols in GF(2^m) can be represented using m bits where each bit is a coefficient in the polynomial $c_{m-1}x^{m-1} + c_{m-2}x^{m-2} + \cdots + c_1x + c_0$. Addition is accomplished using a bit-wise XOR. Multiplication is accomplished by multiplying the corresponding polynomials then reducing the result modulo an irreducible polynomial of degree m. We denote the physical bit separation on the optical medium as τ

2) 2×1 All-Optical Linear Combination Unit Architecture : The 2×1 Linear Combination Unit is a device for computing the linear combination of two input symbols given two scalar coefficients. An overview of the architecture for this unit is shown in Fig. 12.

Because multiplication of the bits representing polynomials results in a product which is twice as big as the operands, we cannot perform serial multiplication entirely in place. In order to free up space on the medium for the extra product bits, we deploy two copies of the multiplication and addition circuit. Two serial inputs A and $B \in$ GF(2^m) are first time-division-demultiplexed onto two physical waveguides using a 1×2 switch which sends adjacent symbols to opposite physical waveguides. Each of the two scalar multiplication and addition units computes the linear combination $C_A \cdot A + C_B \cdot B$ on their respective A and B symbols. The result of each of these operations is a $(2m-1)$-bit symbol. These $(2m-1)$-bit values are normalized to m-bit values concurrently by multiplexing them back onto the same waveguide while simultaneously reducing them modulo an irreducible polynomial in GF(2^m). We will now describe the scalar multiplication/addition unit and the normalization unit each in more detail.

a) 2×1 Scalar Multiplication and Addition Unit: Multiplication in this architecture is accomplished using the shift-and-add approach. Fig. 13 shows this all-optical circuit. The input symbols are split into m copies and then fiber delay lines shift each copy by the appropriate number of bits. The fiber delay lines could be set for a fixed delay or could be controllable to accommodate multiple transmission rates.

With an on-off keying system, each bit of the scalar coefficients $C_A = \langle c_{a,0}, c_{a,1}, \ldots, c_{a,m-1}\rangle$ and $C_B = \langle c_{b,0}, c_{b,1}, \ldots, c_{b,m-1}\rangle$ controls an all-optical "ON/OFF" switch (e.g. an AND gate implemented with an SOA), which effectively multiplies the corresponding copy of the input signal by that bit. A switch in the "OFF" state does not let optical pulses through, effectively transmitting zero. A switch in the "ON" state lets the signal pass through (with possible amplification), effectively transmitting the input symbol times 2^i for coefficient bit $c_{x,i}$. Also, configurable splitters could be used to save on loss due to splitting for each zero bit in the coefficient. Furthermore, if a coding algorithm could limit the number of 1 bits in the coefficient, the size of the splitter and the number of AND gates could be reduced (while possibly increasing the amount of control needed in the delay).

Every symbol on the data stream for a particular connection is multiplied by the same coefficient. Thus, the coefficient-bit control signal is fixed for a particular session, and high speed switching is not necessary. Again, addition in GF(2^m) is merely an XOR of the operands, so it can be accomplished with an all-optical XOR gate. This design uses two 1:m optical splitters, $2m-2$ delay lines (which need not allow the same maximum delay), $2m$ SOAs, and at most $2m-1$ two-input all-optical XOR gates.

This architecture could be used for other keying systems, such as phase-shift keying, so long as the building blocks supported it with the same resulting logical operation. For instance, all-optical XOR gates using return-to-zero differential phase shift keying have been demonstrated [54]. Although, the theory of static network coding assumes the transmission of zeros on failed links. Thus, the XOR would have to treat the absence of a signal the same as a zero, or the device would need to fabricate a stream of zeros in place of the lost signal.

b) 2×1 Serial Normalization Unit Design: We now describe how the result of two different scalar multiplications of m bits can be normalized in GF(2^m) in the same space while multiplexing them back onto the same waveguide.

This normalization unit is shown in Fig. 14. Note that degree of the polynomial after multiplication has degree at most $2m-2$ since $2^{m-1}2^{m-1} = 2^{2m-2}$. Thus, the output of the multiplication unit is $2m-1$ bits. The rectangles labeled $A[2m-2, m]$ and $A[m-1, 0]$ are the upper and lower halves of the output of the first multiplication unit while $B[2m-2, m]$ and $B[m-1, 0]$ represent the output of the second unit. Notice that the lower bits of B will be aligned with the higher bits of A at this point. The normalization unit starts by multiplexing each of these onto two different waveguides so that the top half of each result is sent to the top combiner and the lower half is sent to the lower combiner. The lower bits of B are still aligned with the higher bits of A, but both waveguides will alternate half of A with half of B.

Our design for the normalization part of the circuit is based on the GF(2^m) normalization step in Itoh-Tsujii algorithm [55], [56], [57]. For our circuit, we have selected the reducing polynomial of the form $x^m + x + 1$. This polyno-

	Switch Size	SMUs	Optical Buffers	Adding XORs
full	$WF \times WF$	$(WF)^2$	$(WF)^2$	$(WF)^2 - WF$
limited	$(WF+\beta) \times (WF+\beta\gamma)$	$\beta\gamma$	$\beta\gamma - \beta$	$\beta\gamma - \beta$

TABLE I
A COMPARISON OF THE COMPLEXITY OF THE FULL NETWORK CODING CAPABILITY OXC WITH THE LIMITED CODING CAPABILITY OXC.

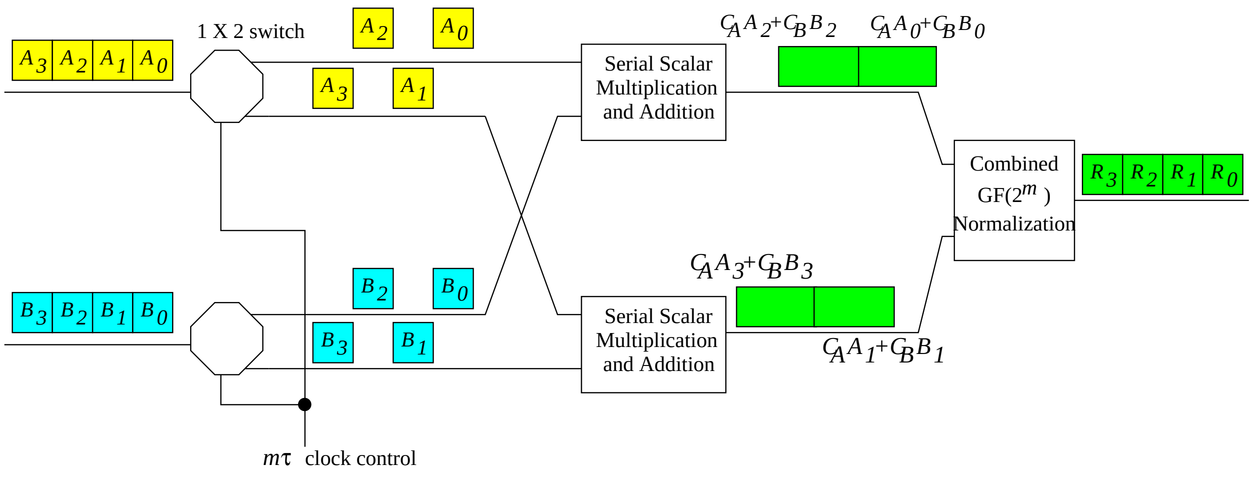

1 X 2 switch
A_3 A_2 A_1 A_0
B_3 B_2 B_1 B_0
Serial Scalar Multiplication and Addition
$C_AA_2+C_BB_2$ $C_AA_0+C_BB_0$
$C_AA_3+C_BB_3$ $C_AA_1+C_BB_1$
Combined GF(2^m) Normalization
R_3 R_2 R_1 R_0
$m\tau$ clock control

Fig. 12. An overview of the architecture for an all-optical coding unit.

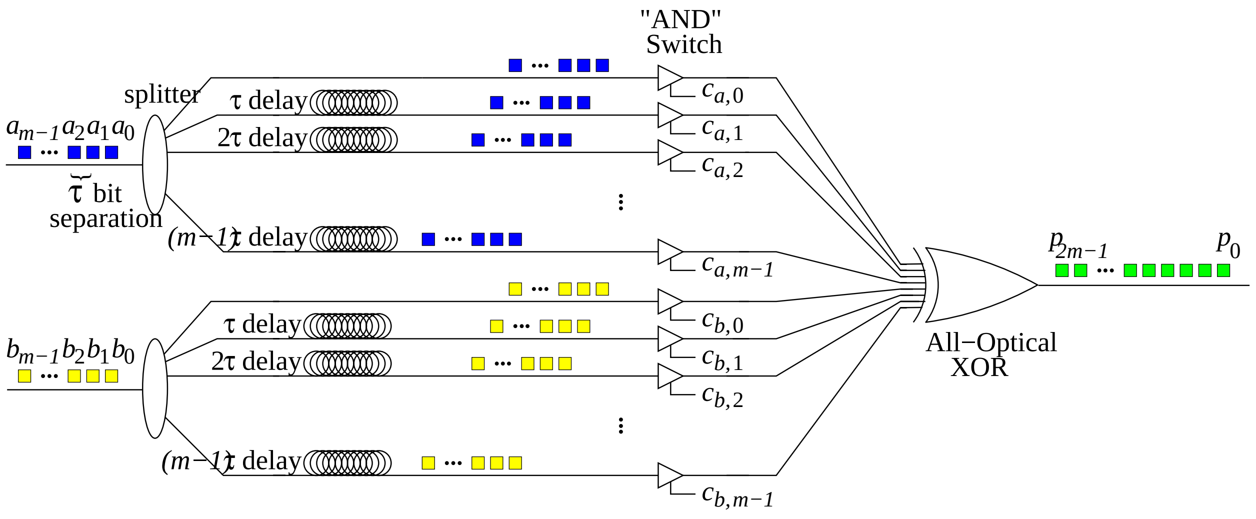


Fig. 13. An all-optical circuit for scalar multiplication and addition in GF(2^m).

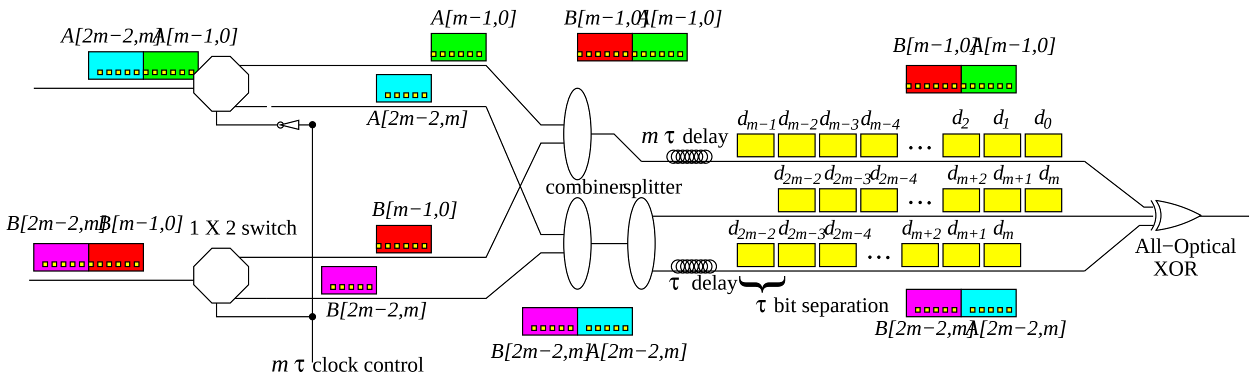


Fig. 14. A serial all-optical circuit for normalization in GF(2^m) when the reducing polynomial is $x^m + x + 1$ (Note: This is only valid for m in which $x^m + x + 1$ is reducible).

mial is irreducible for several choices of m (such as $m = 2, 3, 4, 6, 7, 9, 15, 22, 28, 30, 46, 60$ and 127 [58, p. 158]). This approach reduces a polynomial of the form $d_{2m-2}x^{2m-2} + d_{2m-3}x^{2m-3} + d_{2m-4}x^{2m-4} + \ldots + d_3x^3 + d_2x^2 + d_1x + d_0$ where $d_i \in \text{GF}(2)$ by noticing that

$$\begin{array}{lclcl} x^m & = & x & + & 1 \\ x^{m+1} & = & x^2 & + & x \\ x^{m+2} & = & x^3 & + & x^2 \\ & \vdots & & & \\ x^{2m-4} & = & x^{m-3} & + & x^{m-4} \\ x^{2m-3} & = & x^{m-2} & + & x^{m-3} \\ x^{2m-2} & = & x^{m-1} & + & x^{m-2}. \end{array}$$

That is, the term d_ix^i can be removed for all $i \geq m$ and replaced with two terms in GF(2^m). Thus, the polynomial becomes $r_{m-1}x^{m-1} + r_{m-2}x^{m-2} + r_{m-3}x^{m-3} + \ldots + r_3x^3 + r_2x^2 + r_1x + r_0$ where

$$\begin{array}{lclclcl} r_0 & = & d_0 & \oplus & d_m & & \\ r_1 & = & d_1 & \oplus & d_{m+1} & \oplus & d_m \\ r_2 & = & d_2 & \oplus & d_{m+2} & \oplus & d_{m+1} \\ r_3 & = & d_3 & \oplus & d_{m+3} & \oplus & d_{m+2} \\ & \vdots & & & & & \\ r_{m-3} & = & d_{m-3} & \oplus & d_{2m-3} & \oplus & d_{2m-4} \\ r_{m-2} & = & d_{m-2} & \oplus & d_{2m-2} & \oplus & d_{2m-3} \\ r_{m-1} & = & d_{m-1} & \oplus & & & d_{2m-2}. \end{array}$$

Thus, we may XOR the lower m bits with two copies of the upper $m - 1$ bits (with the second copy being up-shifted one bit) for each of the A and B symbols. This operation is accomplished by splitting the upper half of each symbol onto two space-disjoint waveguides and delaying one of them by 1-bit. The lower half bits are then delayed by m bits so that the lower half of A is aligned with the upper half of A and the lower half of B is aligned with the upper half of B. The yellow boxes labeled $d_0, d_1, \ldots, d_{2m-2}$ show how the bits are aligned for each symbol. The two upper half copies along with the lower half are then XORed to get the normalized value.

This design requires two 1×2 switches, a 1:2 passive splitter, two 2:1 combiners, two fiber delay lines, and two two-input all-optical XOR gates. When coupled with the multiplier unit, the whole device requires $2m$ delay lines and $2m + 1$ two-input XOR gates.

3) Single Scalar Multiplication Unit: The 2×1 Linear Combination Unit allows for two linear combinations which are offset by m bits to be normalized in GF(2^m) simultaneously using the same normalization hardware. That is, it uses one set of normalization hardware to perform the computation

$$R_0 = (C_AA_0 + C_BB_0) \bmod (x^m + x + 1)$$

and outputs a stream of m-bit values $R_0, R_1, R_2, \ldots$. However, it may sometimes be helpful to delay the addition of the two products in order to allow for switch architectures like that in Fig. 10. In this case, we can split the symbols $A_0, A_1, A_2, A_3, \ldots$ onto two physical waveguides with separate scalar multipliers and use the two inputs of the normalization unit to normalize the results of the odd and even indexed multiplications simultaneously. That is, we compute a stream of m-bit values

$$\begin{array}{l} (C_AA_0) \bmod (x^m + x + 1), \\ (C_AA_1) \bmod (x^m + x + 1), \\ (C_AA_2) \bmod (x^m + x + 1), \ldots \end{array}$$

This design functions as the SMU shown in the earlier switch architectures. Such a design allows for more flexibility in placement of network coding devices, however, it may require twice as many normalization units. That is, the SMU-based design uses two extra 1×2 switches, two extra buffers, two extra 2-input XORs, two extra 2:1 combiners, and one extra 1:2 splitter (with two extra amplifiers which are not shown) when compared with the design in the previous section. Thus, it is more efficient to replace pairs of SMUs in the design in Fig. 11 with the 2×1 Linear Combination Unit.

In total, each SMU requires three 1×2 switches, $2m + 2$ SOAs, $2m$ 2-input XORs, and $2m$ buffers. Assuming that the switch in Fig. 11 is constructed as shown in Fig. 2 (including wavelength conversion) and outputs from the coding units need not be switched to the inputs of the coding units, we can compare the complexity of the limited network coding switch with the full network coding switch by updating Table I to include the total number of XORs needed for both multiplication and addition, the number of buffers needed for alignment and processing, and the number of SOAs needed for switching and processing. This comparison of the two approaches is shown in Table II. Note that the limited version reduces the complexity for each of the components proportionally as $\beta\gamma$ varies with respect to $(WF)^2$. Also, note that the complexity increases linearly as m increases. It should be feasible to keep m small since it grows logarithmically with the number of sinks that need to be supported.

4) Parallel Normalization Unit Design: Normalization in GF(2^m) may also take place using a parallel normalization unit. This approach has been proposed for inband-forward-error-correction in SDH/SONET networks [57]. A parallel design requires the use of serial-to-parallel and parellel-to-serial converters in order to perform the processing in a parallel manner while dealing with symbols represented in serial form on the optical medium. Conversion technology for doing this has been proposed for use in label recognition [59], [60], [61]. Using the $x^m + x + 1$ modulus, such a design requires $2m - 1$ two-input XOR gates as well as m splitters but only fixed delay lines. to ensure the proper propagation delay on each of the inputs. Table III gives a summary of this complexity when compared with our proposed design. This comparison also highlights one of the main benefits of our proposed normalization design, which is the level of independence of the complexity based on the symbol size m. The only impact m has on our serial normalization unit design is on the clock speed and the length of one of the delays. The parallel design, on the other hand, has linear complexity in terms of m for the number of amplifiers and XORs needed. The parallel design does not use any optical buffers or 1×2 switches, but given that this technology is needed for alignment and switching anyway (and possibly in the serial-to-parallel conversion device used in the parallel design), our serial design makes a better choice.

	SOAs	**Optical Buffers**	**XORs**
full	$(2m+3)(WF)^2$	$(2m+1)(WF)^2$	$(2m+1)(WF)^2 - WF$
limited (using SMUs)	$(WF)^2 + (2m+2)\beta\gamma + (2\gamma+2)\beta WF$	$(2m+1)\beta\gamma - \beta$	$(2m+1)\beta\gamma - \beta$
limited (using 2×1 LCUs)	$(WF)^2 + 2m\beta\gamma + (2\gamma+2)\beta WF$	$2m\beta\gamma - \beta$	$2m\beta\gamma - \beta$

TABLE II
A COMPARISON OF THE COMPLEXITY OF THE FULL NETWORK CODING CAPABILITY OXC WITH THE LIMITED CODING CAPABILITY OXC IN TERMS OF TOTAL NUMBER OF SOAS, BUFFERS, AND 2-INPUT XOR. WE ASSUME THAT THE β OUTPUTS FROM THE CODING UNITS NEED NOT BE SWITCHED TO THE INPUTS OF THE CODING UNITS.

However, note that m does have an impact on the number of amplifiers, buffers, XORs, and splitters (and therefore loss due to splitting) for our multiplication units.

VI. CONCLUSION

We investigated the algorithmic problem for protected optical-layer multicast connections in this paper. We proposed a heuristic for setting up protected multicast connections using network coding and compared its performance with existing techniques for dedicated multicast protection. We found a remarkable improvement when applying network coding which found valid solutions in every case, even for large multicast groups in which the other algorithms failed to find solutions for more than 80% of the connections. Furthermore, the solutions that were found with other algorithms were much less efficient in terms of bandwidth than the network coding approach. From these results, we can conclude that if network coding services can be provided at the optical-layer, then network coding has the potential to make great strides in the ability to provision protected multicast connections. These connections will have a level of service similar to that of $1+1$ dedicated protection while having bandwidth efficiency paralleling shared multicast protection schemes.

In addition to network coding improving multicast protection performance in the practical setting, we have also shown experimentally that our heuristic finds solutions with cost very close to that of the optimal solutions. Given that our algorithm runs in polynomial time, this shows that network coding protection services can be offered in an efficient manner.

After demonstrating this clear benefit of all-optical network coding, we considered how it could be implemented in an all-optical multicast network. We have presented designs for OXC and all-optical processing unit architectures which enable network coding to be offered at the optical layer using switching components, controllable optical delay buffers, and all-optical XOR gates. Furthermore, we have shown that the network can be configured with as much or as little network coding capability as needed with proportional complexity reductions with respect to reductions in the amount of coding provided. Since these devices rely on some technologies which are still in their infancy, we have given only designs for correct behavior given devices with good signal characteristics. Optimizing these designs based on the operating parameters of specific devices such as power consumption, signal-to-noise ratio, etc. is left as future work.

REFERENCES

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

[2] L. Sahasrabuddhe and B. Mukherjee, "Light trees: optical multicasting for improved performance in wavelength routed networks," *IEEE Commun. Mag.*, vol. 37, no. 2, pp. 67–73, Feb 1999.

[3] D. Thaker and G. N. Rouskas, "Multi-destination communication in broadcast WDM networks: A survey," *Optical Networks Magazine*, vol. 3, no. 1, pp. 34–44, January/February 2002.

[4] Q. Qi, S. Shuangjin, and Q. Kun, "An application of optical multicast technology," in *Communications, Circuits and Systems Proceedings, 2006 International Conference on*, vol. 3, June 2006, pp. 1869–1871.

[5] W. Hu and Q. Zeng, "Multicasting optical cross connects employing splitter-and-delivery switch," *Photonics Technology Letters, IEEE*, vol. 10, no. 7, pp. 970–972, Jul 1998.

[6] M. Ali and J. S. Deogun, "Power-efficient design of multicast wavelength-routed networks," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 10, pp. 1852–1862, October 2000.

[7] M. Ali and J. Deogun, "Cost-effective implementation of multicasting in wavelength-routed networks," *Journal of Lightwave Technology*, vol. 18, no. 12, pp. 1628–1638, Dec 2000.

[8] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371 – 381, February 2003.

[9] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[10] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proceedings of the IEEE International Symposium on Information Theory*, June-4 July 2003, pp. 442–.

[11] 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, June 2005.

[12] D. S. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee, "Achieving minimum-cost multicast: A decentralized approach based on network coding," in *Proc. IEEE Infocom*, vol. 3, March 2005, pp. 1607–1617.

[13] C. Fragouli and E. Soljanin, "Information flow decomposition for network coding," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, March 2006.

[14] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang, *Network Coding Theory*. Hanover, Massachusetts, USA: now Publishers Inc., 2006.

[15] G. Rouskas, "Optical layer multicast: rationale, building blocks, and challenges," *IEEE Netw.*, vol. 17, no. 1, pp. 60–65, Jan/Feb 2003.

[16] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. New York: Plenum, 1972, pp. 85–103.

[17] N. Singhal and B. Mukherjee, "Protecting multicast sessions in wdm optical mesh networks," *Lightwave Technology, Journal of*, vol. 21, no. 4, pp. 884–892, April 2003.

[18] T. Rahman and G. Ellinas, "Protection of multicast sessions in WDM mesh optical networks," in *Optical Fiber Communication Conference. (OFC/NFOEC)*, vol. 2, March 2005.

[19] C. Boworntummarat, L. Wuttisittikulkij, and S. Segkhoonthod, "Light-tree based protection strategies for multicast traffic in transport wdm mesh networks with multifiber systems," in *Communications, 2004 IEEE International Conference on*, vol. 3, June 2004, pp. 1791–1795 Vol.3.

[20] N. Singhal, L. Sahasrabuddhe, and B. Mukherjee, "Provisioning of survivable multicast sessions against single link failures in optical wdm mesh networks," *Journal of Lightwave Technology*, vol. 21, no. 11, pp. 2587–2594, Nov. 2003.

	Amplifiers	**Optical Buffers**	**XORs**	**switches**	**converters**
serial	2	2	2	2	0
parallel	$2m$	0	$2m-1$	0	2

TABLE III
COMPARISON OF THE PROPOSED SERIAL NORMALIZATION UNIT WITH PARALLEL NORMALIZATION. "CONVERTERS" REFERS TO SERIAL TO PARALLEL AND PARALLEL TO SERIAL CONVERTERS EACH OF WHICH HAVE NON-TRIVIAL COST AND COMPLEXITY IN TERMS OF m.

[21] N. K. Singhal, C. Ou, and B. Mukherjee, "Cross-sharing vs. self-sharing trees for protecting multicast sessions in mesh networks," *Comput. Netw.*, vol. 50, no. 2, pp. 200–206, 2006.
[22] A. E. Kamal, "1+N protection in optical mesh networks using network coding on p-cycles," in *Proceedings of IEEE Globecom*, 2006.
[23] ——, "1+N protection against multiple link failures in mesh networks," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Glasgow, Scotland, UK, June 2007.
[24] ——, "GMPLS-based hybrid 1+N link protection over p-cycles: Design and performance," in *Proceedings of IEEE Globecom*, 2007.
[25] ——, "A generalized strategy for 1+N protection," in *Proceedings of the IEEE International Conference on Communications (ICC2008)*, 2008.
[26] A. Kamal and O. Al-Kofahi, "Toward an optimal 1+N protection strategy," in *46th Annual Allerton Conference on Communication, Control, and Computing*, Sept. 2008, pp. 162–169.
[27] O. M. Al-Kofahi and A. E. Kamal, "Network coding-based protection of many-to-one wireless flows," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 5, pp. 787 – 813, 2009.
[28] A. E. Kamal and A. Ramamoorthy, "Overlay protection against link failures using network coding," in *Proceedings of the Conference on Information Sciences and Systems (CISS)*, 2008.
[29] R. Menendez and J. Gannet, "Efficient, fault-tolerant all-optical multicast networks via network coding," in *Conference on Optical Fiber communication/National Fiber Optic Engineers Conference (OFC/NFOEC)*, February 2008, pp. 1–3.
[30] E. D. Manley, J. S. Deogun, and L. Xu, "Network coding for optical-layer multicast," in *Proceedings of the Fifth International Conference on Broadband Communications (Broadnets)*, September 2008.
[31] M. Kim, M. Médard, and U.-M. O'Reilly, "Network coding and its implications on optical networking," in *Optical Fiber Communication Conference*. Optical Society of America, 2009, p. OThO3. [Online]. Available: http://www.opticsinfobase.org/abstract.cfm?URI=URI=OFC-2009-OThO3
[32] Z. Li, B. Li, and L. C. Lau, "On achieving maximum multicast throughput in undirected networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. SI, pp. 2467–2485, 2006.
[33] T. Rasmussen, J. K. Rasmussen, and J. H. Povlsen, "Design and performance evaluation of 1-by-64 multimode interference power splitter for optical communications," *IEEE/OSA Journal of Lightwave Technology*, vol. 13, no. 10, pp. 2069–2074, October 1995.
[34] D. Johnson, K. Hill, F. Bilodeau, and S. Faucher, "New design concept for a narrowband wavelength-selective optical tap and combiner," *Electronics Letters*, vol. 23, no. 13, pp. 668–669, 18 1987.
[35] D. Francis, S. DiJaili, and J. Walker, "A single-chip linear optical amplifier," in *Optical Fiber Communication Conference and Exhibit (OFC)*, vol. 4, 2001, pp. PD13–1–PD13–3 vol.4.
[36] R. Ramaswami and K. N. Sivarajan, *Optical Networks: A Practical Perspective*. San Francisco, California: Morgan Kaufmann Publishers, Inc., 1998.
[37] A. Neukermans and R. Ramaswami, "MEMS technology for optical networking applications," *IEEE Communications Magazine*, vol. 39, no. 1, pp. 62–69, Jan 2001.
[38] D. K. Hunter, M. C. Chia, and I. Andonovic, "Buffering in optical packet switches," *J. Lightw. Technol.*, vol. 16, no. 12, pp. 2081–2094, December 1998.
[39] C. Chang-Hasnain, P.-C. Ku, J. Kim, and S.-L. Chuang, "Variable optical buffer using slow light in semiconductor nanostructures," *Proc. IEEE*, vol. 91, no. 11, pp. 1884–1897, November 2003.
[40] P. Ku, C. Chang-Hasnain, and S. Chuang, "variable semiconductor all-optical buffer," *Elect. Lett.*, vol. 38, pp. 1581–1583, November 2002.
[41] V. P. Heuring, H. F. Jordan, and J. P. Pratt, "Bit-serial architecture for optical computing," *Applied Optics*, vol. 31, no. 17, pp. 3213–3224, 1992. [Online]. Available: http://ao.osa.org/abstract.cfm?URI=ao-31-17-3213
[42] G. Theophilopoulos, K. Yiannopoulos, M. Kalyvas, C. Bintjas, G. Kalogerakis, H. Avramopoulos, L. Occhi, L. Schares, G. Guekos, S. Hansmann, and R. Dall'Ara, "40 ghz all-optical XOR with UNI gate," in *Optical Fiber Communication Conference and Exhibit (OFC)*, vol. 1, 2001, pp. MB2–1–MB2–3 vol.1.
[43] R. Webb, R. Manning, G. Maxwell, and A. Poustie, "40 gbit/s all-optical XOR gate based on hybrid-integrated mach-zehnder interferometer," *Electronics Letters*, vol. 39, no. 1, pp. 79–81, January 2003.
[44] Q. Wang, G. Zhu, H. Chen, J. Jaques, J. Leuthold, A. Piccirilli, and N. Dutta, "Study of all-optical XOR using mach-zehnder interferometer and differential scheme," *IEEE J. Quantum Electron.*, vol. 40, no. 6, pp. 703–710, June 2004.
[45] K. Stubkjaer, "Semiconductor optical amplifier-based all-optical gates for high-speed optical processing," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 6, no. 6, pp. 1428–1435, Nov/Dec 2000.
[46] M. Zhang, L. Wang, and P. Ye, "All optical XOR logic gates: technologies and experiment demonstrations," *IEEE Communications Magazine*, vol. 43, no. 5, pp. S19–S24, May 2005.
[47] E. D. Manley, J. S. Deogun, L. Xu, and D. R. Alexander, "Network coding for WDM all-optical multicast," University of Nebraska – Lincoln, Tech. Rep. TR-UNL-CSE-2009-0007, April 2009.
[48] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for steiner trees in graphs," *Acta Informatica*, vol. 15, pp. 141–145, 1981.
[49] H. Takahashi and A. Matsuyama, "An approximate solution for the steiner problem in graphs," *Math. Jap.*, vol. 24, pp. 573–577, 1980.
[50] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing*. Norwell, MA, USA: Kluwer Academic Publishers, 1998.
[51] M. Ali, *Transmission-Efficient Design and Management of Wavelength-Routed Optical Networks*. Norwell, Massachusetts: Kluwer Academic Publishers, 2001.
[52] B. M. Waxman, "Routing of multipoint connections," *IEEE J. Sel. Areas Commun.*, vol. 6, pp. 1617–1622, December 1988.
[53] Z. Li and B. Li, "Network coding in undirected networks," in *Proceedings of the 38th Annual Conference on Information Sciences*, 2004.
[54] K. Chan, C.-K. Chan, L. K. Chen, and F. Tong, "Demonstration of 20-Gb/s all-optical XOR gate by four-wave mixing in semiconductor optical amplifier with RZ-DPSK modulated inputs," *IEEE Photonics Technology Letters*, vol. 16, no. 3, pp. 897–899, March 2004.
[55] S. Bharathwaj and K. Narasimhan, "An alternate approach to modular multiplication for finite fields GF(2^m) using itoh tsujii algorithm," *The 3rd International IEEE-NEWCAS Conference*, pp. 103–105, June 2005.
[56] T. Itoh and S. Tsujii, "A fast algorithm for computing multiplicative inverses in GF(2^m) using normal bases," *Inf. Comput.*, vol. 78, no. 3, pp. 171–177, 1988.
[57] A. Tychopoulos and O. Koufopavlou, "Optimization of the "FOCUS" inband-FEC architecture for 10-gbps SDH/SONET optical communication channels," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, April 2007, pp. 1–6.
[58] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.
[59] P. C. Sun, Y. T. Mazurenko, W. S. C. Chang, P. K. L. Yu, and Y. Fainman, "All-optical parallel-to-serial conversion by holographic spatial-to-temporal frequency encoding," *Optics Letters*, vol. 20, no. 16, pp. 1728–1730, August 1995.
[60] R. Takahashi, T. Nakahara, H. Takenouchi, and H. Suzuki, "40-gbit/s label recognition and 1×4 self-routing using self-serial-to-parallel conversion," *IEEE Photonics Technology Letters*, vol. 16, no. 2, pp. 692–694, Feb. 2004.
[61] R. Takahashi and H. Suzuki, "1-tb/s 16-b all-optical serial-to-parallel conversion using a surface-reflection optical switch," *IEEE Photonics Technology Letters*, vol. 15, no. 2, pp. 287–289, Feb. 2003.