Restorable Logical Topology in the Face of No or Partial Traffic Demand Knowledge

Reuven Cohen   Gabi Nakibly
Technion – Israel Institute of Technology
Computer Science
Haifa, Israel

***Abstract*—The construction of a logical network on top of a physical (optical) infrastructure involves two intertwined tasks: logical link selection – deciding which pairs of routers will be connected by logical links (lightpaths), and logical link routing – deciding how to route each logical link across the optical network. The operator of such networks is often required to maximize the available throughput while guaranteeing its restorability. This paper is the first to combine these seemingly conflicting goals into one optimization criterion: maximizing the restorable throughput of the end-to-end paths. We address this problem in three cases: when the operator has no knowledge of the future bandwidth demands, when it has partial knowledge, and when it has full knowledge. We present efficient algorithms for each of these cases and use extensive simulations to compare their performance.**

I. Introduction

Modern communication networks consist of a logical topology overlaid on an optical physical infrastructure. Distinguishing between the logical and physical networks is crucial to flexibility and efficiency. However, this distinction gives rise to important cross-layer optimization issues, such as how to guarantee smooth restoration following a failure in the physical network. In this work we study the problem of designing a restorable logical network, which continues to operate efficiently after a physical failure. The input to this problem is a physical (optical) network, which consists of optical switches connected by fiber optic links. Only a subset of those switches has the capability to serve as routers. The logical network that we build consists of routers connected by lightpaths. Each lightpath is established over one or more optical fibers and the optical switches connecting these fibers. The constructed logical network should accommodate the traffic demands not only when all the physical components are operational, but also in the face of a physical failure.

The construction of a logical network is composed of two intertwined tasks: deciding which pairs of routers will be connected by logical links (lightpaths) and deciding how to route each logical link across the optical network. These two tasks are referred to as link selection and link routing, respectively. When setting up optical lightpaths as the links of the logical network, the dominating cost is of the transponders at the two ends of every lightpath, which convert optical to electronic signals and vice versa [12]. Therefore, building a logical network is always subject to a budget constraint, which is translated into an upper bound on the number of lightpaths (logical links) that can be established.

Most past works on designing logical networks assume that the logical links are given, and focus on the link routing task. In contrast, we solve the two tasks together, because they have a tremendous impact on each other. To better understand the link selection and routing problem, consider Figure 1(a). This figure shows a physical network with 16 optical switches connected by 24 optical links. Assuming that only nodes a, b, c and d have the capability to serve as routers, logical links can be established only between pairs of these nodes. Figure 1(b) shows a possible logical network with 4 lightpaths (logical links). This logical network is not resilient because a failure of one node (i or c) or a failure of one logical link ($c-i$ or $i-d$) disconnects node d from the rest of the network. By selecting different logical links, we can get a more resilient network. In Figure 1(c) a failure of node c or link $c-i$ does not disconnect d from the rest of the logical network, but a failure of i or $i-d$ still does. By using the same logical links, but routing the logical link $c-d$ differently, we get the logical network in Figure 1(d), which is resilient to any single physical failure.

To deliver network services with guaranteed Service Level Agreement (SLA), it is not enough for the network operator to create a restorable topology. Very often, the operator should be able to provide "restorable throughput" [4], that is, end-to-end throughput whose availability is guaranteed also in the face of a failure. The desire to guarantee restorable throughput contradicts the desire to maximize throughput availability, because full restoration requires that some bandwidth must be reserved in the event of a failure. Nevertheless, these two requirements can be combined into a single optimization criterion: maximizing the *restorable throughput*. Since failures are often limited to a single network element, it is customary to guarantee restorable throughput under the assumption that a new failure may occur only after the network has recovered from all previous failures.

There are several possible schemes to guarantee end-to-end restorable throughput. These schemes are compared in [4], and the one called "Global Recovery" was shown the be the best. In this scheme, end-to-end logical paths are built over the logical network according to the users' requirements, where

This research was partially funded by the Office of the Chief Scientist of the Israel Ministry of Economy under the Neptune generic research project. Neptune is the Israeli consortium for network programming.

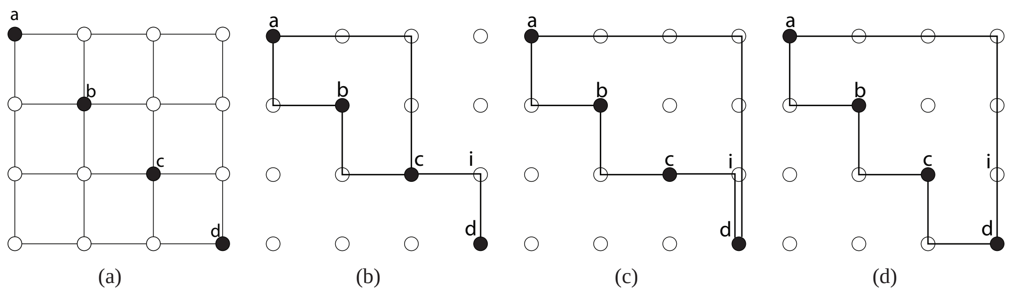


Fig. 1. An example of a physical network (a) and 3 logical networks (b-d) built over the physical network using the same number of links

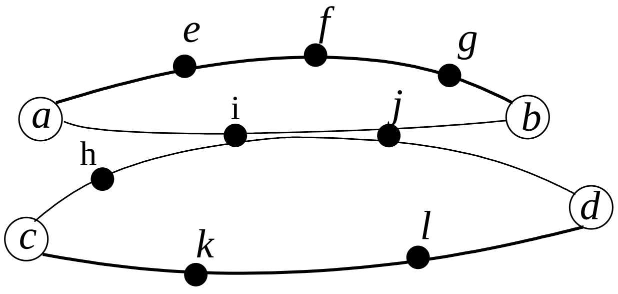


Fig. 2. The global recovery scheme

each logical path consists of one or more logical links. For each logical path, an end-to-end backup path is built in advance between the same pair of end nodes. The backup path protects against all physical link failures along the primary path, with which it shares no physical link.

Figure 2 shows an example of the Global Recovery scheme and the importance of constructing the logical network in such a way as to guarantee end-to-end restorable throughput. There are two primary end-to-end paths (thick lines) with guaranteed restorable throughput of 1Gb/s: $a-e-f-g-b$ and $c-k-l-d$. Note that the links of each path are logical links (lightpaths), built earlier on the optical topology. For each of these primary end-to-end paths, the following backups paths are built: $a-i-j-b$ is the backup path of $a-e-f-g-b$ and $c-h-i-j-d$ is the backup path for $c-k-l-d$. If no pair of the 13 lightpaths (links) shown in this figure shares a physical link, then a failure of one physical link will destroy only one logical link. This implies that a single failure cannot disconnect the two primary end-to-end paths ($a-e-f-g-b$ and $c-k-l-d$) at the same time. Thus, only 1Gb/s on the logical link $i-j$ must be reserved in order to guarantee the availability of 1Gb/s over $a-i-j-b$ if $a-e-f-g-b$ fails and the availability of 1Gb/s over $c-h-i-j-d$ if $c-k-l-d$ fails. In contrast, if the logical links $e-f$ and $k-l$ share an optical link, its failure will destroy both primary end-to-end paths. In such a case, 2Gb/s must be reserved on the logical link $i-j$ for post failure use.

In this paper we propose algorithms for building the logical topology on top of the optical network (i.e., selecting and routing logical links), using a new optimization criterion: maximizing the restorable throughput of the end-to-end paths. We address this problem in the following three cases:

- Case-1: no knowledge of bandwidth demands. Here the operator determines which logical node pairs will be connected by logical links and how each logical link will be routed, when future bandwidth demands are unknown. That is, the operator does not know which primary end-to-end paths will have to be admitted into the network, how many primary paths will be needed, and which nodes are more likely to serve as end points of such paths.
- Case-2: partial knowledge of bandwidth demands. Here, the operator does not know which primary end-to-end paths will have to be admitted into the network, but it knows each node's importance (weight). That is, it knows the relative proportion of traffic expected to originate from or be received by that node.
- Case-3: full knowledge. Here the operator knows which pairs of nodes need to be connected by primary end-to-end paths and the bandwidth demand for each path. While this case only rarely occurs in practice, it can serve us as a benchmark for the other cases.

For lack of information, one cannot formulate an optimization problem that directly maximizes the restorable throughput in the first two cases. Therefore, in these cases we use a the well-known optimization criterion: minimizing the network's *Shared Risk Link Groups* (SRLG). To the best of our knowledge, the earliest reference to the term SRLG is in [26]. However, prior to this, other works addressed the same notion [23], [22]. An SRLG of a physical link e is the set of all logical links routed over e. The cardinality of the SRLG associated with a physical link is known to be a good indicator of the damage to the logical network if this link fails [26]. We seek to minimize the maximum cardinality of the SRLGs over all physical links.

It has not been proved that the SRLG criterion has an unequivocal correlation with restorable throughput. Nevertheless, we choose this criterion because it is widely used in practice, and is widely considered to be a good indicator of logical network resiliency [26].

The main contributions of this paper are as follows:

1) Introducing and formulating the problem of minimizing the maximum SRLG for the selection and routing of logical links.
2) Proposing a near-optimal approximation algorithm for the selection and routing of logical links while minimizing the maximum SLRG when no a-priori information about

the users' bandwidth demands is given (case-1).

3) Proposing a near-optimal approximation algorithm for the selection and routing of logical links while minimizing the maximum SLRG when a-priori information of the nodes' weights is given (case-2).
4) Comparing the restorable throughput admitted in each of the above three cases.

The rest of the paper is organized as follows. Section II discusses related work. In Sections III, IV and V we propose algorithms to construct a logical network for case-1, case-2 and case-3, respectively. In section VI we evaluate the performance of the different algorithms. Finally, Section VII concludes the paper.

II. Related Work

Many works deal with the construction of survivable logical DWDM/MPLS/IP overlay network on top of a physical optical network, e.g., [5], [6], [7], [16], [21]. Most of these works assume that the logical links are given in advance, and focus only on the routing of the logical links subject to some optimization criterion.

To ensure resiliency of the logical network, many network operators seek to minimize the maximum cardinality of the Shared Risk Link Groups (SRLGs) in the logical network [29]. In [21], the authors propose a routing algorithm that maximizes the connectivity of the logical network and propose an integer Linear Program to minimize the maximum SRLG. In [16], the authors study the impact of logical link routing on the amount of spare capacity that should be reserved on the logical links, in order to guarantee the required bandwidth demand also following a single failure. To address this problem, they assume that the bandwidth demand matrix is known in advance. Our work addresses a related problem, i.e., the maximum restorable throughput, while not making this assumption.

The design of a logical network on top of an optical network has been studied extensively for the case where network restoration in not important (e.g., [8], [11], [18] and references therein). Most of these works focus on the efficiency and quality of service of the logical network, without taking into account the possibility of failures.

There are some works that address both the link selection and link routing while taking into account the survivability of the logical network and the traffic it carries. In [13], the authors propose an algorithm to select and route logical links while ensuring that a given traffic demand matrix is satisfied by a set of node-disjoint paths between every pair of logical nodes. In [17], the authors address the problem of logical link selection and routing while minimizing the maximum traffic load on the logical links. Both works assume that the traffic demands are known in advance, in contrast to our work.

Another related line of research which also assumed the existence of a predefined set of lightpath links is one that deals with routing and wavelength assignment problem (see [14], [29], [25] and references therein). In this problem one must route lightpaths over an optical network and assigning a wavelength for each lightpath such as no two lightpaths with the same wavelength traverse the same optical link. In such works the common objective is to maximize the number of routed lightpaths.

Another line of research related to our work is the construction of application level multicast trees (e.g., [3], [15]). Most such works consider a source and a set of destination nodes, and the problem is to construct an overlay tree that satisfies some objective. In [3] and [15], the objective is to minimize the maximum load imposed on the network links, which is equivalent to minimizing the maximum SRLG size. In these works, as in ours, the overlay (logical) nodes are given, while the logical links should be selected and routed. The main difference from our work is that we aim to construct an arbitrary connected logical network, while those works construct a tree.

III. Case-1: No Prior Knowledge of Users' Bandwidth Demands

When no prior knowledge about the end-to-end paths to be established over the logical topology is available, the network opermator can build a resilient logical network by seeking to minimize the maximum cardinality of the Shared Risk Link Groups (SRLGs) in the logical network [29]. As noted above, the cardinality of the SRLG associated with a physical link is known to be a good indicator to the damage caused to the logical network due to a failure of this link. In this section we address for the first time the link selection and routing tasks together, while minimizing the maximum SRLG size. One of our contributions is a near-optimal approximation algorithm that minimizes the maximum SRLG.

A. Problem Definition and Computational Complexity

Let $G_P = (V_P, E_P)$ be an undirected graph that represents the physical (optical) topology, where V_P is the set of optical switches and E_P is the set of optical links. Let $C_p(e)$ be the capacity of $e \in E_P$. Let $V_L \subseteq V_P$ be a subset of the optical nodes that can serve as routers. These routers are the nodes of the logical network, and they are the only physical nodes that can serve as end points of logical links. We assume that the budget allows to establish at most B logical links. All logical links have equal capacities (e.g., OC-12, OC-24,...). We normalize the capacities of the logical links to 1, but multiple logical links can connect a pair of nodes in order to provision higher capacity when needed. We require each logical node to have a degree ≥ 2, in order to ensure minimum resiliency of the logical network. For this requirement to hold, $B \geq |V_L|$ must hold. In this paper we assume that the physical links' capacities are greater than 2 and even. This assumption does not affect the generality of the algorithm, since it holds for all practical optical links (typical values in real-world networks are 40, 80 and 160 [1]).

The problem we define is two-fold. First, we need to decide which pairs of logical nodes should be connected by a lightpath (logical link) while ensuring that the total number of lightpaths is B and the resulting logical network is connected. Second, we need to determine the path over which each of these B logical

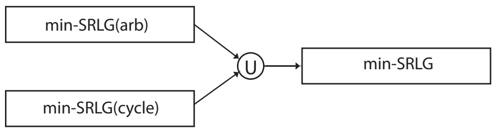


Fig. 3. The structure of the approximation algorithm for MM-SRLG

links should be established.[1]. The goal is to minimize the maximum number of logical links traversing a single physical link. We call this problem *MM-SRLG*.

We first show that MM-SRLG is NP-Complete. To show this, we consider a special case of MM-SRLG where the budget for the logical network allows us to construct only $B = |V_L|$ links. Since the logical graph is required to be connected and the nodes' degrees must be ≥ 2, the graph in this case is actually a simple cycle that spans all logical nodes. Thus, we call this problem *MM-SRLG(cycle)*. The proof that MM-SRLG(cycle) is NP-Complete is provided in Appendix A. Since MM-SRLG(cycle) is a special case of MM-SRLG the following corollary follows.

Corollary 1: MM-SRLG is NP-Complete.

B. An Approximation Algorithm for MM-SRLG

In this section we develop an approximation algorithm for MM-SRLG in three steps. We first present an approximation algorithm for the problem of constructing an arbitrary, not necessarily connected, logical network. Then, we propose another approximation algorithm for solving the problem when the logical network must be a cycle (MM-SRLG(cycle)). Finally, we combine the output of the two algorithms to produce a connected logical graph with the desired number of logical links. The final algorithm guarantees that SRLG $\leq$ OPT+3, where OPT is the optimal solution for MM-SRLG. Figure 3 depicts the structure of the final algorithm.

1) An Algorithm for MM-SRLG(arb): We start with the case were the constructed logical graph is not necessarily connected. We seek to build an arbitrary graph with a predetermined number of logical links, whose SRLG is minimum. We call this problem *MM-SRLG(arb)*. To solve MM-SRLG(arb), consider the reverse variant where the objective is to route the maximum number of logical links between the nodes of V_L such that the number of logical links traversing each physical link e does not exceed $c(e)$, where $c(e)$ is a capacity function on the physical edges. Ref. [9] shows that if $c(e)$ is even for every $e \in E_P$, then there is an integral optimal solution. One can leverage this result and formulate the problem as a straightforward linear program, which can be optimally solved (see Appendix B). Ref. [9] shows that the resulting optimal solution must be integral. We refer to this algorithm as ALG, and propose the following algorithm for the MM-SRLG(arb) problem.

[1]In this work we assume that wavelength conversion is possible in every optical node.

Algorithm 1: (An approximation algorithm for MM-SRLG(arb))

```
1) For $C = 1$ to $\lceil B/2 \rceil$ do (recall that $B$ is the number of logical links)
    a) For every $e \in E_P$, set the capacity of $e$ to be min($2 \cdot C$, $C_p(e)$) and call ALG (i.e., find the logical graph with the maximum number of logical links).
    b) If the number of logical links in the graph is $\geq B$, exit the loop.
2) From the set of logical links, choose an arbitrary subset of size $B$.
```

Each operation of Algorithm 1 runs in polynomial time. Since the algorithm loops at most $B/2$ times, a naive implementation would be pseudo-polynomial. However, the algorithm can be implemented in polynomial time by conducting a binary search on the values of C, instead of running on them sequentially.

Theorem 1: Algorithm 1 builds a logical network whose SRLG $\leq$ OPT+1, where OPT is the SRLG of an optimal solution for MM-SRLG(arb).

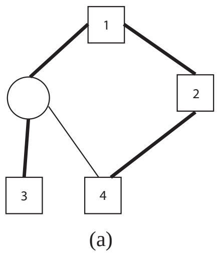
Proof: First, we note that there must be a value of C for which the number of logical links $\geq B$, because the capacity of each physical link may go as high as B. Assume that the algorithm produces a logical network whose maximum SRLG $S' >$ OPT+1. Let C' be the value of C in the final loop of Step 1. We have OPT $\leq S' - 2 \leq 2 \cdot C' - 2 = 2(C' - 1)$. On the other hand, we know that when $C = C' - 1$, the maximum number of logical links is $< B$. Hence, OPT $> 2(C' - 1)$ must hold, which yields a contradiction. ■

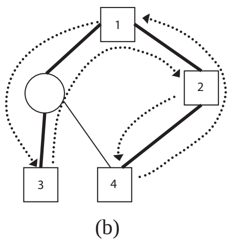
2) An Algorithm for MM-SRLG(cycle): Our next step is to develop an algorithm for MM-SRLG(cycle), which builds a logical cycle whose maximum SRLG does not exceed 2. Since the minimum SRLG for any min-SLRG(cycle) instance is 1, this algorithm can be viewed as a 2-approximation.

Algorithm 2: (An algorithm for a cycle with a SRLG=2)

```
1) Find a tree on $G_P$ that spans $V_L$. This can be done by finding a spanning tree of $G_P$ and then iteratively pruning the leaves that are not in $V_L$, until all the leaves of the tree are in $V_L$.
2) Conduct a DFS tour on the tree, starting at an arbitrary node $v$, to produce a non-simple physical cycle $C$ that traverses all nodes in $V_L$.
3) Break $C$ into sub-paths, such that each sub-path must start and end at logical nodes and each logical node must be an end point of exactly two sub-paths.
4) Transform each sub-path into a logical link with the corresponding routing over the physical graph. Denote this set of logical links as $E_L$.
```

Figure 4 depicts an example of Algorithm 2. Figure 4(a) shows an example of G_P with 4 logical nodes (denoted by rectangles). The bold edges are comprise the tree produced in Step 1. Figure 4(b) shows a cycle produced by a DFS tour. The cycle is divided into 4 sub-paths, each representing a logical link: 1-3, 3-2, 2-4 and 4-1. Figure 4(c) shows the final logical cycle.





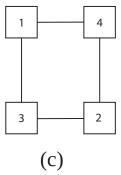


Fig. 4. An illustration of Algorithm 2: (a) a physical graph G_P (logical nodes are denoted by rectangles); (b) a DFS (dashed line) divided into sub-paths; (c) the final logical cycle.

The Algorithm running time is obviously polynomial.

Theorem 2: The logical network generated by Algorithm 2 is a simple cycle whose maximum SRLG is not greater than 2.

Proof: Since the DFS tour traverses all the nodes and returns to the starting node, the constructed logical network is a cycle. Since a logical node is an end point of exactly two links, then the cycle must be simple. The maximum SRLG is not greater than 2 because every link is traversed by the DFS exactly twice (once in each direction). ■

3) The Final Algorithm: We now solve the original min-SLRG problem, which requires the logical graph to be connected and contain no more than B links. To this end, we combine Algorithm 1 and Algorithm 2 in the following way. We first invoke Algorithm 2 to produce a logical cycle. Then, we invoke Algorithm 1 to generate additional logical links, such that their total number will be B.

Algorithm 3: (An approximation for of MM-SRLG)

1) Execute Algorithm 2. Denote its output by E_L^{cycle}.
2) For every $e \in E_P$, reduce $C_p(e)$ by the number of lightpaths in the cycle traversing e.
3) Execute Algorithm 1 with a budget equals to $B' = B - |V_L|$. Denote its output by E_L^{arb}.
4) Return $E_L^{cycle} \cup E_L^{arb}$.

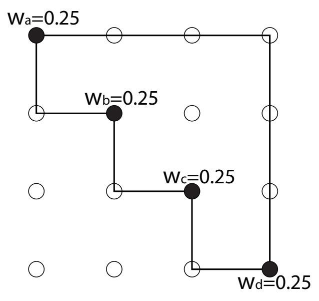
Note that the algorithm may output more than one logical link between a pair of logical nodes. As mentioned in the Introduction, we allow this in our model. This is necessary to accommodate cases in which the network operator wishes to provision between two routers a higher amount bandwidth than a single lightpath allows.

Theorem 3: Algorithm 3 produces a connected logical network whose maximum SRLG $\leq$ OPT+3, where OPT is the value of the optimal solution for MM-SRLG.

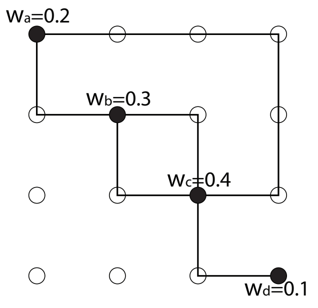
Proof: Since the output graph of Algorithm 2 is a connected logical cycle, the output graph of Algorithm 3 must be connected, and must have a degree of at least 2 for each node. It is easy to see that the number of logical links in the solution returned by the algorithm is B. Let S' denote the maximum SRLG of the solution produced by the algorithm. Consider an arbitrary subset of the logical links in an optimal solution, whose cardinality is $B - |V_L|$. Denote the maximum SRLG of this subset by $S_{B-|V_L|}$. In addition, denote the maximum SRLG of the optimal solution of MM-SRLG(arb) with $B-|V_L|$ logical links by $\text{OPT}_{B-|V_L|}$, and the output of Algorithm 1 (with $B - |V_L|$ logical links) by S_{ALG-1}. Thus,

$$\begin{aligned}\text{OPT} \geq S_{B-|V_L|} \geq \text{OPT}_{B-|V_L|} \geq S_{ALG-1} - 1 \\ \geq (S' - 2) - 1 = S' - 3,\end{aligned}$$

The first inequality holds because $S_{B-|V_L|}$ is the maximum SRLG of a subset of the optimal solution. The second inequality holds because the maximum SRLG of an optimal solution must not be greater than an arbitrary solution, and the third inequality holds due to Theorem 1. Finally, the fourth inequality holds because the maximum SRLG of the cycle produced by Algorithm 2 is 2, which implies that $S' \leq S_{ALG-1} + 2$. ■



(a) a logical network with 4 lightpaths



(b) a logical network with 5 lightpaths

Fig. 5. Two logical networks over a physical network. Option (b) is less robust although it has more logical links

IV. Case-2: Using Information About the Weights of the Logical Nodes

As said earlier, when designing a logical network, the operator is unlikely to know the exact bandwidth demand matrix. However, in many cases each router (logical node) can be associated with a weight, which is proportional to the portion of traffic expected to originate from and received by it. The weights are determined by the network operator before the logical network is set up, based on past experience, on the number of users/subnets expected to be connected through each node, on the importance of the node's geographical location, and so on. Herein we propose to use this weight as an indication of the degree this node should have in the logical network.

The idea that the number of lightpaths connected to every node should be proportional to a weight whose value is known in advance is one of the contributions of this paper. It allows us to build a logical network while taking into account the expected traffic load on each router although the actual traffic matrix is unknown in advance. We illustrate this idea in Figure 5, which shows two logical networks overlayed on the physical network of Figure 1(a).

In Figure 5(a) we assume that the 4 routers are equally important. Thus, each of them has an equal weight of 0.25. In this figure we also assume that the budget of the logical network allows to establish only $B = 4$ lightpaths. Hence, each of the 4 routers should be an end point of $2 \cdot 0.25 \cdot 4 = 2$ lightpaths. Of course, there are other options to establish 4 lightpaths between the 4 nodes such that each node will be connected to 2 lightpaths. In Figure 5(b), the same 4 nodes have different weights and the budget allows to establish $B = 5$ lightpaths. Thus, we have 4 lightpaths connected to node c, 3 to node b, 2 to a and 1 to d. Although there are more lightpaths in Figure 5(b) than in Figure 5(a), the logical network is less robust, because it might be disconnected after a single failure.

We seek to maximize the restorable throughput while taking into account the weight of each logical node. To this end, we propose an approximation algorithm (Algorithm 5) for the minimization of the maximum SRLG while using the weights of the nodes for establishing an upper bound on the degree of the nodes. We show that this algorithm produces logical networks that accommodate more restorable throughput than the networks produced by Algorithm 3. The rationale is that nodes that should accommodate more bandwidth need higher connectivity in order to deliver all the traffic before and after a failure takes place.

A. The Weighted MM-SRLG Problem

In this section we address the weighted variant of MM-SRLG problem. This problem is similar to MM-SRLG, except that the upper bound on the degree of every logical node $v \in V_L$ must be proportional to a given weight w_v which is part of the network. The value of w_v is chosen such that $\sum_{v \in V_L} w_v = 1$.

The degree of each logical node v in G_L is $\leq 2 \cdot B \cdot w_v \cdot \alpha$, where α is an integer ≥ 1. We use α as a parameter that controls the flexibility while setting the degree of each logical node. For $\alpha = 1$, the node degree has no flexibility. As α increases, we allow more flexibility to the degree of each node. This flexibility allows us to obtain a smaller SRLG at the cost of deviating from the nodes' weights. As before, the degree of each logical node is ≥ 2.

In addition to the requirement that $\sum_{v \in V_L} w_v = 1$, we require that for every v, the value of $2B \cdot w_v \alpha$ will be an even integer. While the latter requirement imposes an additional constraint on the w_v vector, we note that: (1) a weight vector that fulfills these two requirements can easily be found using an integer program whose input is the original w'_v vector that fulfills only the $\sum_{v \in V_L} w'_v = 1$; the linear program minimizes the total difference between w_v an w'_v; (2) for every v, the maximum difference between its original weight w'_v and the ultimate weight w_v is smaller than $\frac{1}{2B \cdot \alpha}$, which in practice would be less than 1% since typically $B > 50$ in most real-world networks; (3) the weight vector serves only as a rough estimate of the importance of each node, and the small (approximately 1%) "rounding error" that we add keeps it as such.

We develop an approximation algorithm for the weighted MM-SRLG problem in a similar way to what we did in Section III for MM-SRLG. We first develop an algorithm for the weighted MM-SRLG(arb) problem, where the constructed graph is not necessarily connected. Then, we solve the MM-SRLG(cycle) problem where a logical cycle over the physical graph is produced. Finally, we combine the solutions of the two algorithms into a solution for the general problem.

1) An Algorithm for Weighted MM-SRLG(arb): In the weighted MM-SRLG(arb) problem we do not require the logical graph to be connected, but we take into account the weight constraints on the logical degrees. Note that since no connectivity constraint is imposed on the logical graph, the following algorithm does not assume that $B \geq |V_L|$. To take the weights into account, we transform the physical graph G_P into $G'_P = (V'_P, E'_P)$ as follows. First, for every logical node $v \in V_L$ we define a mirror node v'. We denote the set of logical nodes in the new graph as V'_L. The set V'_L consists only of mirror nodes. Then $V'_P = V_P \cup V'_L$ and $E'_P = E_P \cup \{(v, v') | v \in V_L\}$. This means that each mirror node is connected to its original node. The weight of every mirror node $v' \in V'_L$ is $w_{v'} = w_v$.

Algorithm 4: (An approximation for weighted MM-SRLG(arb))

```
1) Construct $G'_P$ and $V'_L$ according to the above transformation.
2) For every $v \in V_L$, set the capacity of the link $(v, v')$ to be $2 \cdot B \cdot w_v \cdot \alpha$.
3) For $C = 1$ to $\lceil B/2 \rceil$ do
    a) Set the capacity of every $e \in E_P$ to be min($2 \cdot C$, $C_p(e)$).
    b) Call ALG to find the maximum set of logical links in $G'_P$ while using $V'_L$ as logical nodes.
    c) If the total number of logical links is $\geq B$, then select an arbitrary subset of these links of size $B$, and exit the loop.
4) For each logical link in the transformed graph that connects a pair of mirror nodes, add to $E_L$ a logical link in the original graph which connects the nodes that are attached to these mirror nodes.
```

Theorem 4: Algorithm 4 builds a logical network whose SRLG $\leq$ OPT+1, where OPT is the SRLG of an optimal solution for weighted MM-SRLG(arb).

The proof of this theorem is similar to the one presented for Theorem 1. The graph transformation used in the above algorithm has no impact on the approximation factor, since the SRLG on the links of the form (v, v') is not taken into account in the final solution.

2) The Final Algorithm: We are now ready to present the final approximation algorithm for the weighted MM-SRLG problem, which combines Algorithm 4 and Algorithm 2 (presented in Section III-B2).

Algorithm 5: (an approximation Algorithm for weighted MM-SRLG)

```
1) Execute Algorithm 2. Denote its output by $E_L^{cycle}$.
2) For every logical node $v$, set its new weight to $\frac{w_v - 1/B}{1 - |V_L|/B}$ (to offset the new degree of each node, which is now set to 2).
3) For every $e \in E_P$, reduce $C_p(e)$ by the number of lightpaths in the cycle traversing $e$.
4) Execute Algorithm 4 with a budget that equals $B' = B - |V_L|$. Denote its output by $E_L^{arb}$.
5) Return $E_L^{cycle} \cup E_L^{arb}$.
```

Theorem 5: Algorithm 5 produces a logical network whose maximum SRLG $\leq$ OPT+3, where OPT is the value of the optimal solution of weighted MM-SRLG which is not greater than the optimal solution plus 3.

Proof: Since the output graph of Algorithm 2 is a cycle, the output of Algorithm 5 must be connected. It is easy to see that the number of logical links in the solution returned by the algorithm is B. E_L^{cycle} imposes on each node a degree of 2. E_L^{arb} imposes on each node a degree $\leq 2 \cdot B' \cdot \frac{w_v - 1/B}{1 - V_L/B} \cdot \alpha$. Consequently, each node in the final logical network has a link degree $\leq 2 + 2 \cdot B' \cdot \frac{w_v - 1/B}{1 - V_L/B} \cdot \alpha < 2 \cdot B \cdot w_v \cdot \alpha$.

The proof for the approximation bound is similar to the one presented in the proof of Theorem 3. Let S' denote the maximum SRLG of the solution produced by the algorithm. Consider an arbitrary subset of the logical links in an optimal solution, whose cardinality is $B - (|V_L|)$. Denote the maximum SRLG of this subset by $S_{B-|V_L|}$. In addition, denote the maximum SRLG of the optimal solution of MM-SRLG(arb) with $B - |V_L|$ logical links by $\text{OPT}_{B-|V_L|}$, and the output of Algorithm 4 (with $B - |V_L|$ logical links) by S_{ALG-4}. Thus,

$$\begin{aligned} \text{OPT} \geq S_{B-|V_L|} &\geq \text{OPT}_{B-|V_L|} \geq S_{ALG-4} - 1 \\ &\geq (S' - 2) - 1 = S' - 3, \end{aligned}$$

The first inequality holds because $S_{B-|V_L|}$ is the maximum SRLG of a subset of the optimal solution. The second inequality holds because the maximum SRLG of an optimal solution must not be greater than an arbitrary solution, and the third inequality holds due to Theorem 4. Finally, the fourth inequality holds because the maximum SRLG of the cycle produced by Algorithm 2 is 2, which implies that $S' \leq S_{ALG-4} + 2$. ∎

V. Case-3: Assuming Full Knowledge of The Users' Bandwidth Demands

We now address case-3, where the operator knows the average bandwidth demands between every pair of nodes on the logical network. In this case, we solve a joint optimization problem that:

1) Selects the logical links and routes them on the physical network.
2) Determines the end-to-end primary and backup paths for all traffic demands, and the bandwidth reserved on each path.

Case-3 only rarely occurs in practice, because the network operator only rarely has concrete knowledge of the future bandwidth demands when the logical network is constructed. Still, this case can serve as a benchmark for the maximum restorable throughput in case-1 and case-2. The joint optimization problem for this case can be formulated as an integer linear program (ILP), which we solve by relaxing its integrality constraints.

The linear program for the global recovery scheme requires as an input the primary path of each traffic flow. However, we do not have this information when the logical network is constructed. We solve this problem by building a backup path for every flow and every logical link the flow traverses, rather than building a single backup path for every flow. This strategy requires every backup path to start and end at the nodes of the failed logical link, and it is slightly inferior to the global recovery scheme [4].

The linear program has the following parameters:

- F – the set of all logical node pairs. Each pair $f = (s_f, t_f) \in F$ constitute a flow.
- d_f – the bandwidth demand of flow $f \in F$. This value is normalized to the capacity of a single logical link; namely, $d_f = 1$ means that the bandwidth demand equals the capacity of a logical link.
- B – the total number of logical links that are budgeted to the network.

We define the following variables:

- $y^e_{f,(u,v)}$ – the fraction of d_f routed over the logical edge connecting the logical nodes u and v when physical edge $e \in E_P$ fails; when no edge fails, $e = \phi$.
- x_f – the total routed fraction of d_f.
- $l_{(u,v)}$ – an integer variable that equals the number of logical links connecting the two logical nodes u and v.
- $r_e^{(u,v)}$ – an integer variable that equals the number of logical links connecting the two logical nodes u and v and that traverse the physical link $e \in E_P$.
- $\overline{r}_e^{(u,v)}$ – a binary variable that equals 1 if and only if $r_e^{(u,v)} \geq 1$.

In the following $l_{(u,v)}$ represents a directed link. However, we treat each such directed link as an undirected link with the corresponding capacity. For example, suppose that $l_{(2,1)} = 2$, $l_{(1,2)} = 2$ and $l_{(3,1)} = 1$. Then, there are 5 undirected links connected to node '1', 4 undirected links connected to node '2' and 1 undirected link connected to node '3'.

The target function is to maximize the total throughput $\sum_f d_f \cdot x_f$, subject to the following constraints:

(1) $\sum_{e=(j,i)\in E_P} r_e^{(u,v)} - \sum_{e=(i,j)\in E_P} r_e^{(u,v)}$

$$= \begin{cases} -l_{(u,v)} & i = u \\ l_{(u,v)} & i = v \quad ,\forall i \in V_P, \forall u,v \in V_L \\ 0 & \text{else} \end{cases}$$

(2) $\sum_{u,v\in V_L} r_e^{(u,v)} \leq C_p(e),\ \forall e \in E_P$

(3) $\sum_{u,v\in V_L} l_{(u,v)} = B$

(4) $\sum_{u\in V_L} y^e_{f,(u,v)} - \sum_{u\in V_L} y^e_{f,(v,u)}$

$$= \begin{cases} x_f & v = t_f \\ -x_f & v = s_f \quad ,\forall v \in V, \forall f \in F, \forall e \in \{E_P, \phi\} \\ 0 & \text{else} \end{cases}$$

(5) $\sum_{f\in F} d_f \cdot y^e_{f,(u,v)} + \sum_{f\in F} d_f \cdot y^e_{f,(v,u)} \leq l_{(u,v)} + l_{(v,u)}$
$\forall u,v \in V_L, \forall e \in \{E_P, \phi\}$

(6) $y^e_{f,(u,v)} \leq 1 - \overline{r}_e^{(u,v)}$ $,\forall f \in F, \forall e \in E_P, \forall u,v \in V_L$

(7) $\overline{r}_e^{(u,v)} + y^e_{f,(u,v)} \geq y^\phi_{f,(u,v)}$ $,\forall f \in F, \forall e \in E_P, \forall u,v \in V_L$

(8) $r_e^{(u,v)} \leq \overline{r}_e^{(u,v)} \cdot B$ $,\forall e \in E_P, \forall u,v \in V_L$

(9) $\sum_{u\in V_L} l_{(u,v)} + l_{(v,u)} \geq 2$ $,\forall v \in V_L$

(10) $0 \leq x_f \leq 1,\ 0 \leq y^e_{f,(u,v)} \leq 1$
$\forall f \in F, \forall u,v \in V_L, \forall e \in \{E_P, \phi\}$

(11) $\overline{r}_e^{(u,v)} \in \{0,1\}$, $r_e^{(u,v)} \in \{0,1,2,\ldots\}$,
$l_{(u,v)} \in \{0,1,2,\ldots\}$
$\forall e \in E_P, \forall u,v \in V_L.$

The set (1) of constraints ensures that every logical link is routed over the physical link. Note that in this ILP, the physical links must be directed. Therefore, E_P represents here a set of directed edges, where each undirected link is represented by a pair of oppositely directed links. Constraint (2) ensures that the capacities of the physical links are not exceeded. Constraint (3) ensures that the total number of logical links is B. The set (4) of constraints ensures the conservation of traffic over the logical network. The set (5) ensures that traffic flows only over the selected logical links. Note that $l_{(u,v)}$ represents the number of directed logical links from u to v. Since in our problem a logical link is undirected, each chosen logical link gives bandwidth to both directions. Therefore, in the right-hand side of each inequation in (5) we sum the number of logical links from both directions of each pair of logical nodes. The set (6) ensures that traffic flow will not be carried by logical links that traverse a failed physical link. The set (7) ensures a traffic flow will be carried by the same logical links for all link failure events unless the logical link traverses a failed physical link. The set (8) ensures a proper relation between $r_e^{(u,v)}$ and $\overline{r}_e^{(u,v)}$; namely, $\overline{r}_e^{(u,v)}$ is set to 1 if $r_e^{(u,v)}$ is greater than 0. The set (9) ensures that all logical nodes have a degree of at least 2. The set (10) of constraints ensures that the total routed bandwidth of each flow do not exceed flow demand. Finally, the set (11) of constraints ensures that each logical link is routed over a single physical path.

The above program is an integer linear program (ILP). To allow tractability in the simulation study we solve the fractional version of this ILP where we relax the integrality constraints. This relaxation allows us to get an upper bound to the solution of case-3. This solution will serve as a benchmark for the solutions of case-1 and case-2.

VI. Simulation Study

In this section we evaluate the performance of the proposed algorithms using extensive simulations. We first evaluate the SRLG performance of Algorithms 3 (case-1) and 5 (case-2). Then, we compare the restorable throughput in the three cases. Throughout this section we use $\alpha = 2^2$ in Algorithm 5.

A. SRLG Performance

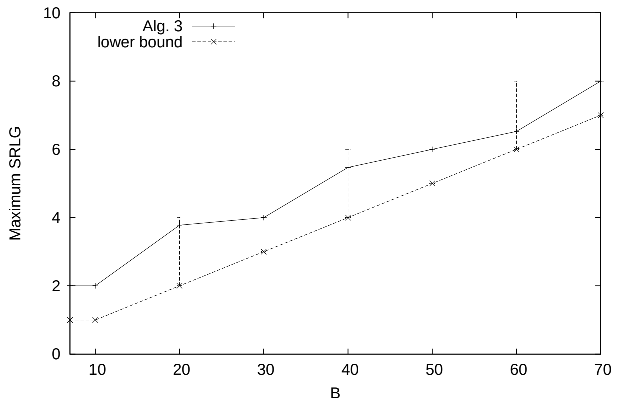
We first evaluate the SRLG performance of Algorithm 3. As in earlier related work [20], [21], [28], we use augmented real network topologies as our physical network: the NSFNET topology, which has 14 nodes and 29 links, and the USIP topology, which has 24 nodes and 53 links. For each topology we generate 125 random instances, each comprises of a subset of physical nodes, which serve as logical routers, and a budget B of logical links. For simplicity, and to allow faster execution of the linear program, we do not impose a constraint on the capacities of the physical links. For each instance we ran the following algorithms:

1) Algorithm 3: This is the MM-SRLG approximation algorithm presented in Section III-B3.
2) An algorithm that finds a lower bound (i.e. the best possible result) for MM-SRLG. This algorithm finds an optimal solution for the splittable variant of the problem, which can be represented as a linear problem and be solved in polynomial time. Obviously, an optimal solution to this variant is always better than or equal to the optimal solution for MM-SRLG. We then round up the fractional optimal solution returned by the algorithm. The rounded result is still a lower bound for MM-SRLG, because the optimal solution for MM-SRLG is also integral. The linear program is presented in Appendix C.

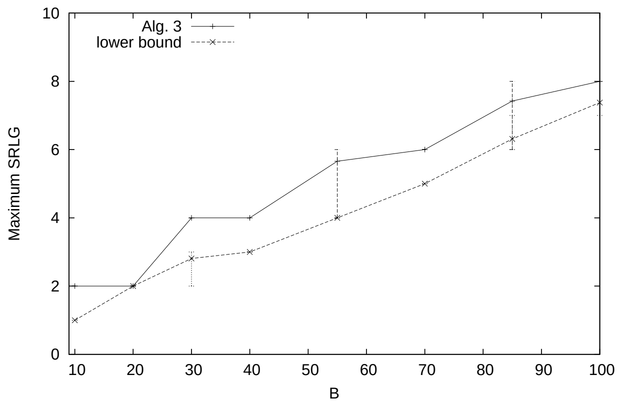
In each simulation instance, the locations of the logical routers are randomly chosen using a uniform distribution. The number $|V_L|$ of logical routers is chosen as either $1/3$ or $1/2$ of the number of physical nodes $|V_P|$.The budget of logical links (B) varies between $|V_L| + 2$ and $14 \cdot |V_L|$.

Figure 6 shows the simulation results for the two physical topologies. In all the graphs the x-axis represents the budget B of logical links, while the y-axis represents the maximum SRLG size produced by each algorithm. For each point in the

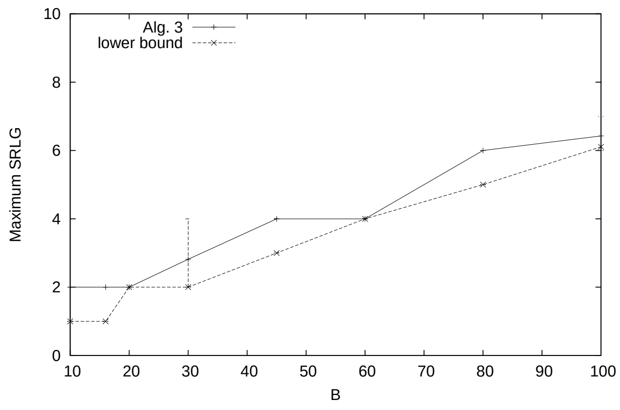
[2]Following some initial simulations, we observed that the best performance is obtained with $\alpha = 2$. Nonetheless, we cannot claim that the same value of α is the best choice for every system.



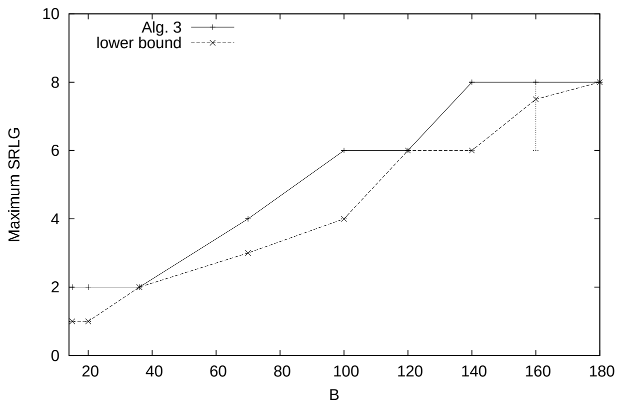
(a) NSFNET physical topology, $|V_L|$=5



(b) NSFNET physical topology, $|V_L|$=7



(c) USIP physical topology, $|V_L|$=8



(d) USIP physical topology, $|V_L|$=12

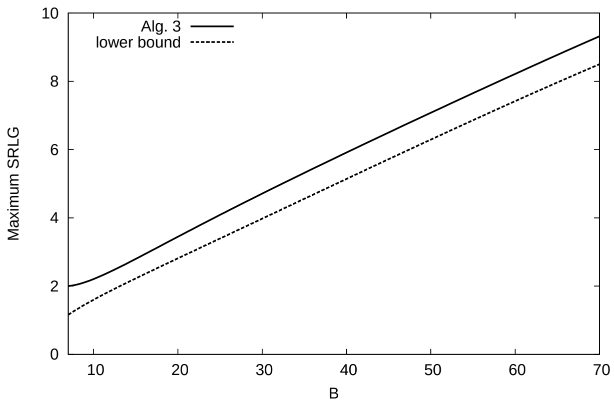
Fig. 6. The SRLG performance (with error bars) of Alg. 3 compared to the theoretical lower bound

graphs, we show an error bar for the range of its y values. Each bar depicts the distance between the minimum value of the "maximum SRLG" and the maximum value of the "maximum SRLG" produced by different simulation instances for a given B value. In each such a simulation instance, the locations of logical nodes are different, because they are randomly chosen according to uniform distribution. For most of the points there was no difference between the minimum and maximum "maximum SRLG" produced by the different simulation instances, and therefore there is no error bar. We see in all graphs that Algorithm 3 performs very well, and it is very close to the theoretical lower bound. No error bar for most of the points means that, for most budgets, the maximum SRLG size does not depend on the location of the chosen nodes. This indicates that the maximum SLRG size is influenced mainly by the number of logical links (B) and much less by the locations of the logical nodes.

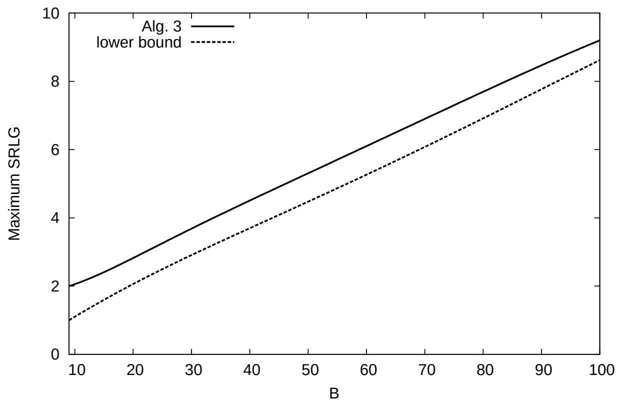
It is evident from all the graphs that given a number of logical nodes, the maximum SRLG increases with the budget B. However, when the number of logical nodes increases, then for the same number of logical links the maximum SRLG size decreases to some extent. The reason is that by increasing the number of nodes, one has more flexibility in choosing and routing the logical links. Finally, by comparing the performance of the algorithms for the two physical topologies, we observe that for the same V_L/V_P and E_L/E_P ratios, the performance of the algorithms in the USIP topology degrade to some extent. This can be explained by the fact that the average length of the physical route of each logical link is longer for larger physical topologies, which imposes higher load on the network.

To validate the above results we depict in Figure 7 the performance of the above two algorithms while using artificially generated physical topologies. We randomly generate 20 physical topologies having 15 nodes and 30 links (equivalent in size to the NSFNET topology). To generate these topologies we use the BRITE simulator [24] while employing the Barabasi-Albert model [2]. This model captures two important characteristics of network topologies: incremental growth and preferential connectivity of a new node to well-connected existing nodes. These characteristics yield a power-law degree distribution of the nodes. As for NSFNET we constructed logical networks with 5 and 7 nodes in Figure 7(a) and Figure 7(b), respectively.

The results of Figure 7 closely resemble the ones for the NSFNET topology. Here also the difference between Algo-



(a) $|V_L|$=5



(b) $|V_L|$=7

Fig. 7. The SRLG performance of Alg. 3 compared to the theoretical lower bound averaged over 20 artificially generated physical topologies

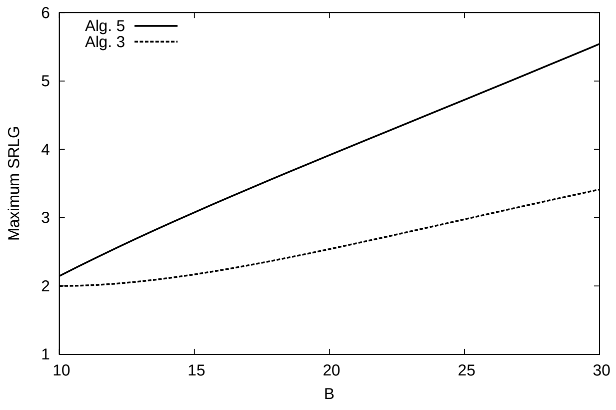


Fig. 8. The SRLG performance of the algorithms given NSFNET as the physical topology, and $|V_L| = 7$

rithm 3 and the theoretical lower bound is less than 1. The curves are smoother than the curves for NSFNET, because we average the results over many instances of physical topologies.

We now evaluate the SRLG performance of Algorithm 5. The evaluation is done using the same simulation setup described above. We only show here the results for the NSFNET physical topology. Figure 8 shows the SRLG for Algorithm 5 and Algorithm 3. The x-axis represents the budget B of logical links, while the y-axis represents the maximum SRLG achieved by each algorithm. It is evident that Algorithm 3 obtains the lowest maximum SLRG size, which is not surprising since this algorithm is not restricted by the node weights.

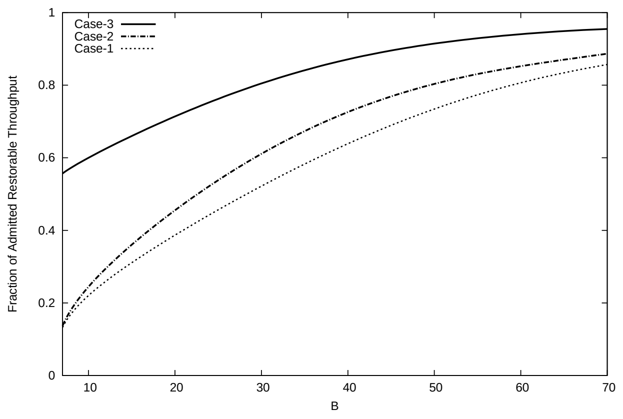
B. Restorable Throughput Performance

We now evaluate the restorable throughput admitted by the logical networks constructed in each of the three cases. To measure this criterion, we generate traffic demands for the logical network based on the Gravity model [19], where the demand of every pair of logical nodes is proportional to the products of their weights. We remind the reader that all logical links have the same capacity, and that bandwidth demands are normalized by the capacity of the logical link. To measure the maximum restorable throughput that can be admitted into the logical network, we use the optimal algorithm presented in [4] for the Global recovery scheme [27]. As discussed earlier and depicted in Figure 2, in this scheme each primary logical path is associated with one backup logical path between the same end nodes. Each backup path protects against the failure of any physical link in the primary path, and it therefore does not share any physical link with this path. This scheme is sometimes referred to as "path recovery scheme."

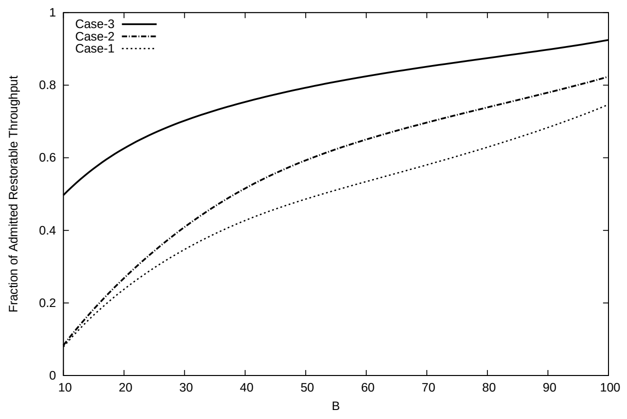
Figure 9 depicts the restorable throughput for the constructed logical network with 5 or 7 logical nodes. For this simulation, we use the NSFNET as the physical topology. The x-axis represents the number of logical links, denoted B, whereas the y-axis represents the fraction of traffic demand that can be admitted into the logical network while guaranteeing 100% restorability in the face of a single physical failure. As expected, the fraction of admitted throughput increases with B for all the algorithms. Furthermore, the algorithm for case-3 obtains up to 120% better throughput than the other two, with its advantage depending on the number of logical links. As this number increases, the advantage of the case-3 algorithm decreases due to the greater flexibility in the selection of the logical links, which in turn makes the full knowledge of future demands less important.

The difference between the curves of case-1 and case-2 is interesting. We can see that case-2 has better restorable throughput than case-1 (up to 25%) for all B values. This result indicates the importance of knowing the nodes' weights in advance. While we obtained similar results in other scenarios, we cannot claim, of course, that this will always be the case.

Figure 10 depicts the performance of case-2 as a function of the α-value used during Algorithm 5. As before, NSFNET is the physical topology. This time we use $V_L = 7$. The x-axis represents the value of α, whereas the y-axis represents the restorable traffic demand of case-2 normalized by the restorable traffic demand of case-1. We consider 6 different scenarios



(a) $|V_L|$=5



(b) $|V_L|$=7

Fig. 9. The restorable throughput performance of the algorithms given NSFNET as the physical topology

with different values of B: 30, 40, 50, 60, 80 and 100. It is evident from the graph that for low B-values up to 50, the best performance is obtained for $\alpha = 1$, while for higher values, $\alpha = 2$ is optimal. This suggests that for low B-values it is more important that the network will be designed according to the expected traffic distribution rather than according to only SRLG considerations.

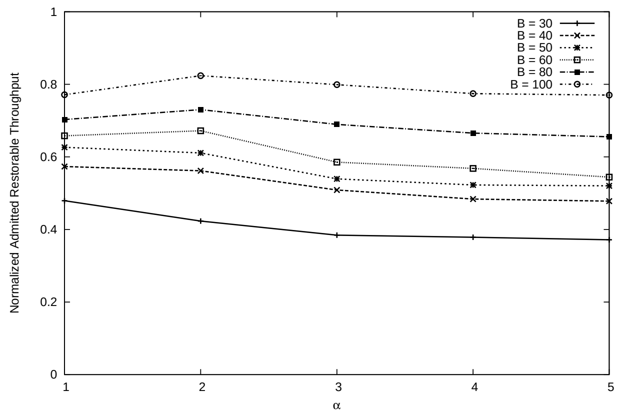


Fig. 10. The normalized restorable throughput as a function of α. $|V_L| = 7$

VII. Conclusion

This paper is the first to propose algorithms for building the logical topology on top of the optical network while maximizing the restorable throughput of the end-to-end paths. We addressed this problem in three cases: when the operator has no knowledge of the future bandwidth demands, when it has partial knowledge, and when it has full knowledge. For the first case the most reasonable strategy is to minimize the maximum SRLG size. We showed that the resulting (new) problem is NP hard, and proposed an efficient approximation for solving it. For the second case we assumed that each router (logical node) can be associated with a weight, which is proportional to the portion of traffic expected to originate from and received by it. We used this weight as an indication of the degree this node should have in the logical network. In the third case, we assumed that the operator knows the average bandwidth demands between every pair of nodes on the logical network. We presented efficient algorithms for each of these cases, and ran extensive simulations to compare their performance.

References

[1] Dwdm - dense wavelength division multiplexing. http://www.fiberoptic.com/adt_dwdm.htm.

[2] A. Barabasi, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the World Wide Web. In *Physica A: Statistical Mechanics and Its Applications*, volume 281, pages 69–77, June 2006.

[3] R. Cohen and G. Kaempfer. A unicast-based approach for streaming multicast. In *Proceedings of IEEE INFOCOM*, volume 1, pages 440–448, 2001.

[4] R. Cohen and G. Nakibly. Maximizing restorable throughput in MPLS networks. *IEEE/ACM Transactions on Networking*, 18(2):568–581, 2010.

[5] O. Crochat and J.-Y. Le Boudec. Design protection for WDM optical networks. *IEEE Journal on Selected Areas in Communications*, 16(7):1158–1165, September 1998.

[6] O. Crochat, J.-Y. Le Boudec, and O. Gerstel. Protection interoperability for WDM optical networks. *IEEE/ACM Transactions on Networking*, 8(3):384–395, June 2000.

[7] Q. Deng, G. Sasaki, and C. fong Su. Survivable IP Over WDM: An efficient mathematical programming problem formulation. In *IEEE HSN*, 2002.

[8] R. Dutta and G. N. Rouskas. A survey of virtual topology design algorithms for wavelength routed optical networks. *Optical Networks*, 1:73–89, 2000.

[9] A. Frank, A. V. Karzanov, and A. Sebo. On integer multiflow maximization. *SIAM J. Discret. Math.*, 10:158–170, 1997.

[10] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., 1979.

[11] A. Gençata and B. Mukherjee. Virtual-topology adaptation for WDM mesh networks under dynamic traffic. *IEEE/ACM Trans. Netw.*, 11(2):236–247, April 2003.

[12] O. Gerstel, R. Ramaswami, and G. H. Sasaki. Cost-effective traffic grooming in WDM rings. *IEEE/ACM Transactions on Networking*, 8:618–630, 2000.

[13] L. Gouveia, P. Patrcio, and A. de Sousa. Hop-constrained node survivable network design: An application to mpls over wdm. *Networks and Spatial Economics*, 8:3–21, 2008.

[14] B. Jaumard, C. Meyer, and B. Thiongane. Comparison of ILP formulations for the RWA problem. *Optical Switching and Networking*, 4(34):157 – 172, 2007.

[15] X. Jin, W. Yiu, S. Chan, and Y. Wang. On maximizing tree bandwidth for topology-aware peer-to-peer streaming. *IEEE Transactions on Multimedia*, 9:1580–1592, 2007.

[16] D. Kan, A. Narula-Tam, and E. Modiano. Lightpath routing and capacity assignment for survivable IP-over-WDM networks. In *DRCN*, Oct. 2009.

[17] G. R. Kiran and C. S. R. Murthy. QoS based survivable logical topology design in WDM optical networks. *Photonic Network Communications*, 7:193–206, 2004.

[18] V. R. Konda and T. Y. Chow. Algorithm for traffic grooming in optical networks to minimize the number of transceivers. In *IEEE Workshop on High Performance Switching and Routing*, pages 218–221, 2001.

[19] J. Kowalski and B. Warfield. Modeling traffic demand between nodes in a telecommunications network. In *Proceedings of ATNAC*, 1995.

[20] K. Lee, H.-W. Lee, and E. Modiano. Reliability in layered networks with random link failures. In *Proceedings of the 29th conference on Information communications*, INFOCOM, pages 1667–1675, 2010.

[21] K. Lee and E. Modiano. Cross-layer survivability in WDM-based networks. *IEEE/ACM Transactions on Networking*, 2011.

[22] D. Medhi. A unified approach to network survivability for teletraffic networks: Models, algorithms and analysis. *IEEE Trans. on Communications*, 42:534–548, 1994.

[23] D. Medhi and S. Sankarappan. Impact of a transmission facility link failure on dynamic call routing circuit-switched networks under various circuit layout policies. *Journal of Network and Systems Management*, 1:143–169, 1993.

[24] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An approach to universal topology generation. In *Proceedings of MASCOTS*, 2001.

[25] A. E. Ozdaglar and D. P. Bertsekas. Routing and wavelength assignment in optical networks. *IEEE/ACM Transactions on Networking*, 11(2):259–272, April 2003.

[26] B. Rajagopalan, D. Pendarakis, D. Saha, R. Ramamoorthy, and K. Bala. IP over optical networks: architectural aspects. *IEEE Communications Magazine*, 38(9):94–102, September 2000.

[27] V. Sharma and F. Hellstrand. Framework for multi-protocol label switching (MPLS)-based recovery. IETF RFC 3469, February 2003.

[28] T. Venkatesh, T. Sujatha, and C. Murthy. A novel burst assembly algorithm for optical burst switched networks based on learning automata. *Optical Network Design and Modeling*, 4534:368–377, 2007.

[29] H. Zang and J. P. Jue. A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks. *Optical Networks Magazine*, 1:47–60, 2000.

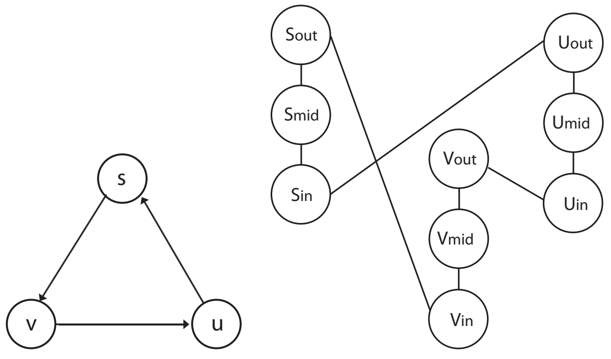
APPENDIX A
CASE-1: MM-SRLG(CYCLE) IS NP-COMPLETE

We prove that MM-SRLG(cycle) is NP-Complete using a reduction from the Directed Hamiltonian Cycle (DHC) problem [10]. The reduction is similar to the one presented in [3]. In the DHC problem, a directed graph $G(V,E)$ is given, and the problem is to determine if G contains a directed Hamiltonian cycle (i.e., a simple cycle that passes through all the vertices of V).

Theorem 6: MM-SRLG(cycle) is NP-Complete.

Proof: MM-SRLG(cycle) is clearly in NP. We now show a reduction from DHC. Given a directed graph $G(V,E)$ as an instance of DHC, we construct an instance of MM-SRLG(cycle) and show that the DHC instance contains an Hamiltonian cycle if and only if the constructed MM-SRLG(cycle) instance contains a logical network whose maximum SRLG is 1. We define the undirected physical graph $G_P = (V_P, E_P)$ as follows:

$$V_P = \left\{ v^{in}, v^{mid}, v^{out} | v \in V \right\},$$



(a) A given instance of DHC (b) The reduced MM-SRLG(cycle) instance

Fig. 11. An illustration of a reduction from an instance of DHC to an instance of MM-SRLG(cycle)

$$E_P = \left\{ (v^{in}, v^{mid}), (v^{mid}, v^{out}) | v \in V \setminus \{s\} \right\} \\ \cup \left\{ (v^{out}, u^{in}) | (v,u) \in E \right\}.$$

In addition, we choose the following set of logical nodes:

$$V_L = \left\{ v^{mid} | v \in V \setminus \{s\} \right\}.$$

Figure 11 depicts an example of the above reduction. If G contains a directed Hamiltonian cycle C, this cycle can be transformed into the following set of logical links:

$$E_L = \left\{ (u^{mid}, v^{mid}) | (u,v) \in C \right\},$$

where (v^{mid}, u^{mid}) is a logical link routed over the physical path $(v^{mid}, v^{out}, u^{in}, u^{mid})$. Since a logical link was constructed for every edge in C, then the logical network must form a cycle. By the construction, since for every $v \in V$ there is exactly one edge in C that enters v and exactly one edge in C that exits v, then every edge in E_P is used by at most one logical link. Thus, (V_L, E_L) is a logical graph having SRLG=1.

To prove the other direction, we need to show that if the created MM-SRLG(cycle) instance contains a logical cycle whose SRLG is 1, then the original DHC instance contains a directed Hamiltonian cycle. We denote the logical cycle formed by E_L as C_L. By definition, E_L is a set of edge-disjoint physical paths. Thus, and by construction, every vertex $v^{mid} \in V_L$ has at most two logical edges attached to it. We construct a cycle C in G by choosing an edge (u,v) for each logical link $(u^{mid}, v^{mid}) \in E_L$. Since C_L is a cycle that connects all the logical vertices in V_L, C must be a Hamiltonian cycle. ■

APPENDIX B
THE LINEAR PROGRAM FOR ALG

We present the linear program used for ALG in Algorithms 1 and 4. The target function of the program is maximizing the number of logical links, given a capacity constraint $c(e)$ for every $e \in E_P$:

Maximize L

subject to the following constraints:

$$
\begin{aligned}
&(1) \quad \textstyle\sum_{e=(j,i)\in E_P} r_e^{(u,v)} - \sum_{e=(i,j)\in E_P} r_e^{(u,v)} \\
&\quad = \begin{cases} -l_{(u,v)} & i=u \\ l_{(u,v)} & i=v \quad ,\forall i\in V_P, \forall u,v\in V_L \\ 0 & \text{else} \end{cases} \\
&(2) \quad \textstyle\sum_{u,v\in V_L} l_{(u,v)} = L \\
&(3) \quad \textstyle\sum_{u,v\in V_L} r_e^{(u,v)} \le c(e) \ ,\forall e\in E_P \\
&(4) \quad r_e^{(u,v)} \le 1,\ l_{(u,v)} \le 1 \\
&\quad \forall e\in E_P, \forall u,v\in V_L.
\end{aligned}
$$

The set (1) of constraints ensures that every logical link is routed over a physical link. Constraint (2) ensures that the total number of logical links is L. The set (3) of constraints ensures that the capacity constraints of the physical links are not violated.

Appendix C
The Linear Program for The SRLG Lower Bound

In this section we present the linear program to find a lower bound of the optimal maximum SRLG. This program is the fractional version of the following ILP that finds the optimal SRLG.

The target function of the program is to minimize the maximum cardinality of the SLRGs:

Minimize $\quad z$

subject to the following constraints:

$$
\begin{aligned}
&(1) \quad \textstyle\sum_{e=(j,i)\in E_P} r_e^{(u,v)} - \sum_{e=(i,j)\in E_P} r_e^{(u,v)} \\
&\quad = \begin{cases} -l_{(u,v)} & i=u \\ l_{(u,v)} & i=v \quad ,\forall i\in V_P, \forall u,v\in V_L \\ 0 & \text{else} \end{cases} \\
&(2) \quad \textstyle\sum_{u,v\in V_L} l_{(u,v)} = B \\
&(3) \quad \textstyle\sum_{u,v\in V_L} r_e^{(u,v)} \le z \ ,\forall e\in E_P \\
&(4) \quad r_e^{(u,v)} \in \{0,1,2\ldots\}\ ,\ l_{(u,v)} \in \{0,1,2\ldots\} \\
&\quad \forall e\in E_P, \forall u,v\in V_L.
\end{aligned}
$$

The set (1) of constraints ensures that every logical link is routed over the physical link. Constraint (2) ensures that the total number of logical links is B. The set (3) of constraints ensures that the variable z takes the maximum cardinality of the all SRLGs. Finally, the set (4) of constraints ensures that each logical link is routed over a single physical path.

PLACE PHOTO HERE

Reuven Cohen received the B.Sc., M.Sc. and Ph.D. degrees in Computer Science from the Technion - Israel Institute of Technology, completing his Ph.D. studies in 1991. From 1991 to 1993, he was with the IBM T.J. Watson Research Center, working on protocols for high speed networks. Since 1993, he has been a professor in the Department of Computer Science at the Technion. He has also been a consultant for numerous companies, mainly in the context of protocols and architectures for broadband access networks. Reuven Cohen has served as an editor of the IEEE/ACM Transactions on Networking and the ACM/Kluwer Journal on Wireless Networks (WINET). He was the co-chair of the technical program committee of Infocom 2010 and headed the Israeli chapter of the IEEE Communications Society from 2002 to 2010.

PLACE PHOTO HERE

Gabi Nakibly received the B. Sc. in Information Systems engineering (summa cum laude) and PhD in Computer Science from the Technion - Israel Institute of Technology, Haifa, Israel, in 1999 and 2008, respectively. Gabi is a professional fellow in the National Research & Simulation Center at Rafael Advanced Defense Systems. He also serves as an adjunct researcher and lecturer in the Computer Science department at the Technion. Gabi received his Ph.D. in computer Science in 2008 from the Technion, and he is a recipient of the Katzir Fellowship. His main research interests include network security and traffic engineering.