Computer Networks 74 (2014) 1–12

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Network protection with multiple availability guarantees ☆,☆☆

Greg Kuperman [a,*], Eytan Modiano [a], Aradhana Narula-Tam [b]

[a] Massachusetts Institute of Technology, LIDS, Cambridge, MA 02139, USA
[b] MIT Lincoln Laboratory, Lexington, MA 02420, USA

ARTICLE INFO

Article history:
Received 23 August 2013
Received in revised form 6 August 2014
Accepted 19 August 2014
Available online 27 August 2014

Keywords:
Network reliability
Network protection
Network recovery
Network design

ABSTRACT

This paper develops a novel network protection scheme that provides guarantees on both the fraction of time a flow has full connectivity, as well as a quantifiable minimum grade of service during downtimes. In particular, a flow can be below the full demand for at most a maximum fraction of time; if after a network failure the flow is below its full demand, it must still support at least a fraction q of that demand. This is in contrast to current protection schemes that offer either availability-guarantees with no bandwidth guarantees during the downtime, or full protection schemes that offer 100% availability after a single link failure.

We show that the multiple availability guaranteed problem is NP-Hard, and develop an optimal solution in the form of an MILP. If a connection is allowed to drop to 50% of its bandwidth for just 1 out of every 20 failures, then a 24% reduction in spare capacity can be achieved over traditional full protection schemes. Allowing for more frequent drops to partial flow, additional savings can be achieved. Algorithms are developed to provision resources for connections that provide multiple availability guarantees for both the sharing and non-sharing case. For the case of $q = 0$, corresponding to the standard availability constraint, an optimal pseudo-polynomial time algorithm is presented.

1. Introduction

As data rates continue to rise, a network failure can cause catastrophic service disruptions. To protect against such failures, networks typically use full protection schemes, often doubling the cost of resources needed to route a connection. An alternative approach is to provide a guarantee on the maximum time a connection can be disrupted. This is known as an "availability guarantee", and it is a bound on the fraction of time or probability that a connection can be disrupted. However, these disruptions (downtimes) may be unacceptably long; thus, many service providers opt for the more resource intensive full protection. In this paper, we propose a novel protection scheme with multiple availability guarantees. In addition to the traditional availability guaranteed protection, which maintains the full demand for at least a guaranteed fraction of time, we guarantee partial connectivity at all times. Thus, our approach is a hybrid between the traditional availability guarantees and full protection schemes.

Full protection schemes have been studied extensively [1–7]. The most common full protections schemes are $1+1$ or $1:1$ guaranteed path protection [8]. In $1+1$ path protection, two copies of the data are sent over a primary

☆ Preliminary version of this work appeared in the 2012 IEEE International Conference on Communications (ICC).

☆☆ This work was supported by NSF Grants CNS-1116209 and CNS-0830961, by DTRA Grants HDTRA1-07-1-0004 and HDTRA-09-1-005, and by the Department of the Air Force under Air Force contract #FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

* Corresponding author.
E-mail addresses: gregk@mit.edu (G. Kuperman), modiano@mit.edu (E. Modiano), arad@ll.mit.edu (A. Narula-Tam).

http://dx.doi.org/10.1016/j.comnet.2014.08.013
1389-1286/

path and a failure disjoint protection path. Since two copies of the data are sent, the connection is guaranteed to survive any individual path failure. The downside of this strategy is that the protection resources are always utilized, and cannot be used to protect another connection while the original primary path is functioning. Alternatively, $1:1$ protection reserves resources on a disjoint backup path for protection, but does not utilize that path until a failure has occurred. With proper sharing strategies, protection resources can be used by multiple primary demands as long as they are not needed for more than one connection at any given moment in time. The disadvantage of $1:1$ protection is the additional complexity required for implementation. In this paper, we will refer to both the $1+1$ and $1:1$ protection schemes as disjoint path full protection, and specify the particular form as needed.

In addition to full protection schemes, there has also been a growing body of literature for backup provisioning to meet availability guarantees [9–15]. In all of these, primary and backup flows are allocated such that the connection is disrupted for at most a specified fraction of time or probability. During these down-states, the service is completely disrupted. A version of availability guarantees is considered in [16], where an end-to-end flow having a certain expected capacity, based on link availabilities, is found; multi-path routing is used to distribute risk, but no guarantees on flow are provided. In our paper, a flow is guaranteed to be at least a fraction q of the full demand at all times, which is known as "partial protection". Our novel approach is the first to combine the traditional availability guarantee and partial protection guarantee to allow the user to specify flows with different availability guarantees. Moreover, it is particularly applicable to IP-over-WDM networks where MPLS tunnels are used to provision resources.

The partial protection framework was first introduced in [17]. More recently, [18,19] developed a "theory" of partial protection such that after *any* single link failure, the flow can drop to the partial protection requirement. In [18,19], a fraction q of the demand is guaranteed to remain available between the source and destination after any single link failure, where q is between 0 and 1. When q is equal to 1, the service will have no disruptions after any single failure, and when q is 0, there will be no flow between the two nodes during the down state. In our work, flows can drop below the full demand for at most a specified fraction of time, and maintain at least q of that demand at all times.

The novel contributions of this paper include a framework for Multiple Availability Guaranteed Protection (MAGP) and providing associated algorithms to provision resources to meet these guarantees for both the cases when protection resources can and cannot be shared. Moreover, in the $q=0$ case, corresponding to the previously studied scenario where full availability is guaranteed for a fraction of time, we develop an optimal pseudo-polynomial algorithm. A preliminary version of this work was published in [20].

This paper is outlined as follows. In Section 2, the model for MAGP is described. In Section 3, MAGP is shown to be NP-Hard, and the minimum-cost solution to MAGP is formulated as an MILP. In Section 4, optimal solutions and algorithms for MAGP are developed when protection resources cannot be shared, and in Section 5, an algorithm is developed for when protection resources can be shared.

2. Multiple availability guaranteed protection

In this paper, routing strategies are developed and analyzed to minimize the total cost and capacity allocation required to satisfy each demand's protection and availability requirements. A demand needs to be routed from its source s to destination t such that the flow must be fully available for some given percentage of time. In other words, a flow can drop below the full demand for at most some specified downtime for any given time period, and must maintain at least a fraction q of that full demand at all times. Primary and protection resources are provisioned at the time of routing for a connection, which guarantees that sufficient capacity exists after a failure for that flow to meet its availability requirements. Similar to [11–15], the probability of simultaneous failures is assumed to be negligible, and we only consider single-link failures. To simplify the analysis, a "snapshot" model is used: The network state is considered after a failure has occurred. Let p_{ij} be the conditional probability that edge $\{i,j\}$ failed given that a network failure has occurred. For ease of exposition, instead of availability or maximum downtime, the Maximum Failure Probability (MFP) is considered, and its value is denoted by P. After some network failure occurs, a flow can be below the full demand, but at least a fraction q of the demand, with at most probability P. When a flow is below its full demand (but always at least q), that connection will be considered in a "downstate". The maximum failure probability is the conditional probability that a connection is in a downstate given some link disruption has occurred in the network.

This maximum failure probability can be related to the metric of availability by accounting for the expected time between failures and mean time to repair. Assuming that both the time between failures and the length of repair of any failure as exponential random variables with parameters $\frac{1}{\lambda}$ and $\frac{1}{\mu}$, respectively. The expected proportion of time there will be some failure is $\frac{\mu}{\lambda+\mu}$. With MAGP, after some failure in the network, a connection can fall below its full demand with probability less than P. In other words, $(1-P)$ percent of failures must have no effect (i.e., zero repair time). With a maximum failure probability P, the expected value for repair time becomes μP, and the proportion of time a connection is down is $\frac{\mu P}{\lambda+\mu P}$. We note that with MAGP, when a connection is "down", it still maintains a fraction q of the original connection's demand.

We assume that the graph G, with a set of vertices V, edges E, and edge failure probabilities $\mathcal{P}$, is at least two-connected. Since only single-link failures are considered, edge failures are disjoint events; hence, the sum of all the link failure probabilities is equal to one (i.e., $\sum_{\{i,j\}\in E} p_{ij} = 1$). Similar to previous works, the primary flow is restricted to a single path. After the failure of a link, a network management algorithm reroutes the traffic along the allocated protection paths.

Consider the example in Fig. 1, with link failure probabilities and flow allocations as labeled (p and f respectively). A unit demand needs to be routed from s to t with $P = \frac{1}{4}$ and partial protection requirement q. In [18], a simple partial protection scheme called $1+q$ protection was developed, which routes the primary demand on one path and the partial protection requirement onto another edge-disjoint path. After any failure along the primary path, the partial protection requirement is met. This is shown in Fig. 1a with the solid line carrying the primary flow of 1 and the dotted line carrying the protection flow of q. However, in this example the maximum failure probability is exceeded for the $1+q$ routing: after a failure, the flow drops below the unit demand between s and t with a probability of $\frac{1}{2}$ (because the failure of either of the primary links would drop the demand below its full capacity). A naive alternative would be to simply allocate another path for protection, which would be similar to the $1+1$ (or $1:1$) full protection scheme (shown in Fig. 1b), and utilize 4 units of capacity. After any failure, the full demand of 1 is maintained; thus, the user will face no downtime, which meets and exceeds the maximum probability of failure requirement of $\frac{1}{4}$.

If we allow different levels of protection on different segments of the primary path, then a more resource efficient allocation is possible. Consider keeping the primary flow on the same bottom two edges as before, but instead of allocating an end-to-end backup path along the top two edges, 1 unit of flow is allocated to protect against the failure of $\{s, v\}$ and q units of flow to protect against the failure of $\{v, t\}$ (shown in Fig. 1c). If after some disruption either of the $\{s, v\}$ edges fail, 1 unit of flow will still remain from s to t. By fully protecting the primary $\{s, v\}$ edge, there is zero probability that its failure will cause the flow to drop below the full demand. The probability that the flow will drop below 1 after some failure is $\frac{1}{4}$, which meets the MFP requirement. This routing only needs $3+q$ units of capacity, as opposed to the 4 units that full protection requires.

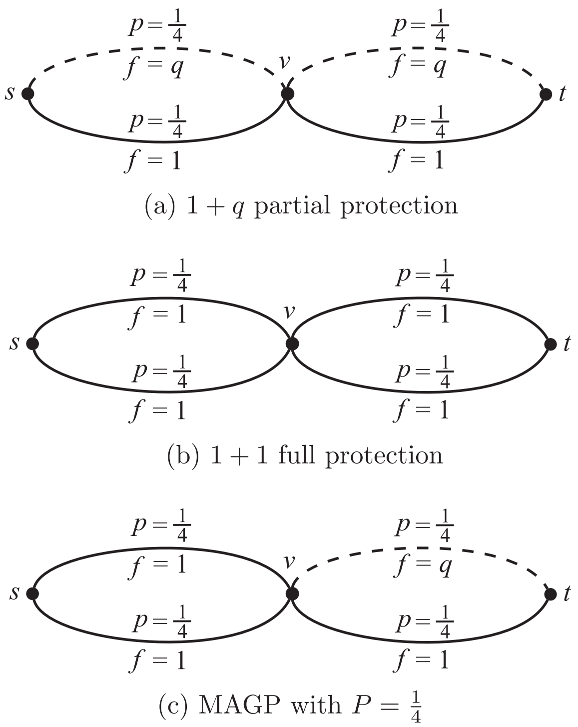

Fig. 1. Comparison of MAGP and traditional protection schemes.

3. Minimum-cost multiple availability guaranteed protection

This section investigates minimum-cost allocations for Multiple Availability Guaranteed Protection (MAGP). We first define the MAGP problem. Then, the MAGP problem is shown to be NP-Hard. Subsequently, in Section 3.1 an MILP is formulated to find a minimum-cost routing that meets a demand's partial protection and availability requirements. In Section 3.2, MAGP is compared to the guaranteed $1+1$ or $1:1$ full protection scheme.

We assume a graph G, with a set of vertices V, edges E, and edge failure probabilities $\mathcal{P}$. Each edge $\{i,j\}$ has an associated cost c_{ij}. After a single-link failure, the network can enter a downstate with at most a Maximum Failure Probability (MFP) of P, where the MFP is the conditional probability that the network is in a downstate given some link disruption. During a downstate, the flow between a source and destination may fall below the full demand, but must always remain at a minimum fraction q of the demand. We now show that finding the minimum-cost solution to MAGP is NP-Hard.

Theorem 1. *The minimum-cost multiple availability guaranteed protection problem is NP-Hard.*

Proof. To demonstrate NP-Hardness of MAGP, a reduction from the 1–0 knapsack problem [21] is performed. See Appendix A for the complete proof. □

3.1. Mixed integer linear program to meet multiple availability guaranteed protection

Since finding a minimum-cost solution for MAGP is NP-Hard, in this section a mixed integer linear program (MILP) is developed to solve for the minimum-cost solution. For a connection request between two nodes s and t, the flow can drop to a fraction q of the demand with at most probability P. Again, the snapshot model is used, and the set of link failure probabilities $\mathcal{P}$ are conditional given a network failure has occurred.

For the MILP, the following values are given:

- $G = (V, E, C, \mathcal{P})$ is the graph with its set of vertices, edges, costs, and edge failure probabilities, respectively.
- d^{st} is the required demand between nodes s and t.
- q^{st} is the fraction of the demand between s and t that must be supported in the event of a link failure.
- c_{ij} is the cost of link $\{i,j\}$.
- p_{ij} is the probability that link $\{i,j\}$ has failed given a network failure has occurred.

- P^{st} is the maximum probability that the service between s and t falls below its full demand after some network failure.

The following variables will be solved:

- x_{ij}^{st} is the primary flow for demand (s,t) on link $\{i,j\}, x_{ij}^{st} = \{0,1\}$.
- w_{ij} is the total primary flow assigned on link $\{i,j\}, w_{ij} \geqslant 0$.
- s_{ij} is the spare capacity assigned on link $\{i,j\}, s_{ij} \geqslant 0$.
- z_{kl}^{st} is 1 if the failure of link $\{k,l\}$ causes the flow between s and t to drop below the full demand for that connection; 0 otherwise.
- $f_{ij,kl}^{st}$ is the flow on link $\{i,j\}$ after the failure of link $\{k,l\}$ for demand $(s,t), f_{ij,kl}^{ij} \geqslant 0$.
- $y_{ij,kl}^{st}$ is the spare capacity on link $\{i,j\}$ for failure of link $\{k,l\}$ for demand $(s,t), y_{ij,kl}^{st} \geqslant 0$.

Objective:

- Minimize the cost of allocation over all links.

$$\text{minimize} \sum_{\{i,j\}\in E} c_{ij}(w_{ij} + s_{ij}) \tag{1}$$

Subject to:

- Flow conservation constraints for primary flow: route primary traffic to meet the set of demands.

$$\sum_{\{i,j\}\in E} x_{ij}^{st} - \sum_{\{j,i\}\in E} x_{ji}^{st} = \begin{cases} 1 & \text{if } i=s \\ -1 & \text{if } i=t \\ 0 & \text{otherwise} \end{cases}, \quad \forall i \in V, \ \forall (s,t) \in (V,V) \tag{2}$$

- Full demand availability constraint: The probability that the flow between s and t drops below 1 after a failure is simply the sum of the failure probabilities of the individual edges causing the flow to drop below 1. The sum of these failure probabilities cannot exceed P.

$$\sum_{\{k,l\}\in E} p_{kl} z_{kl}^{st} \leqslant P^{st}, \quad \forall (s,t) \in (V,V) \tag{3}$$

- Flow conservation constraints for partial service: if the failure of link $\{k,l\}$ causes the flow to drop below the full flow for demand (s,t), route q^{st} from s to t; otherwise, maintain the full flow. Let $\mathcal{F}_{kl}^{st}$ be the expression $(1 - z_{kl}^{st}) + q^{st} z_{kl}^{st}$.

$$\sum_{\substack{\{i,j\}\in E \\ \{i,j\}\neq\{k,l\}}} f_{ij,kl}^{st} - \sum_{\substack{\{j,i\}\in E \\ \{j,i\}\neq\{k,l\}}} f_{ji,kl}^{st} = \begin{cases} \mathcal{F}_{kl}^{st} & \text{if } i=s \\ -\mathcal{F}_{kl}^{st} & \text{if } i=t \\ 0 & \text{otherwise} \end{cases}, \quad \forall i \in V, \ \forall (s,t) \in (V,V), \forall \{k,l\} \in E \tag{4}$$

- Working allocation is enough on link $\{i,j\}$ for all demands.

$$\sum_{(s,t)\in(V,V)} d^{st} x_{ij}^{st} = w_{ij}, \quad \forall \{i,j\} \in E \tag{5}$$

- Capacity allocation: primary and spare capacity assigned on link $\{i,j\}$ meets protection requirements after the failure of link $\{k,l\}$.

$$f_{ij,kl}^{st} \leqslant x_{ij}^{st} + y_{ij,kl}^{st}, \quad \forall \{i,j\} \in E, \ \forall \{k,l\} \in E, \ \forall (s,t) \in (V,V) \tag{6}$$

- Spare allocation is enough on link $\{i,j\}$ for all demands after failure of edge $\{k,l\}$

$$\sum_{(s,t)\in(V,V)} d^{st} y_{ij,kl}^{st} \leqslant s_{ij}, \quad \forall \{i,j\} \in E, \ \forall \{k,l\} \in E \tag{7}$$

For some demand between nodes s and t, a minimum-cost solution will provide an edge capacity allocation such that the flow drops to a fraction q^{st} of that demand with at most probability P^{st}.

3.2. Comparison to full protection

Multiple availability guaranteed protection is compared to disjoint path full protection via simulation. The performance of the strategies is compared using the NSFNET topology (Fig. 2) with 100 random unit demands. The protection requirement q is set to $\frac{1}{2}$ for all demands. All link costs are set to 1, and the probability of failure for any link is proportional to its length, which is reasonable since a longer fiber will have a higher likelihood of being accidently cut. The maximum failure probability P is varied from 0 to .3 by .05 increments. While the main focus of this paper is the case where the primary flow is restricted to a single path, this simulation also considers allowing the primary flow to be bifurcated. Bifurcation reduces the loss of flow after any edge failure, thereby reducing the total allocation needed to meet requirements. Relaxing the binary integer constraint on the primary flow variables in the MILP corresponds to enabling bifurcation of the primary flow. Routing solutions for MAGP were determined using CPLEX to solve the MILP. Due to the length of running time of the MILP, each demand is run individually, which is the case without protection resource sharing. When resources cannot be shared, $1+1$ and $1:1$ protection will become the same in terms of resource utilization. Protection resource sharing is considered in Section 5, where a one-at-a-time routing scheme is utilized. The shortest pair of disjoint paths were used for $1+1$ protection [22].

The average cost to route the demand and protection capacity using the different routing strategies is plotted in Fig. 3 as a function of the maximum failure probability P. The shortest path routing without protection consider-

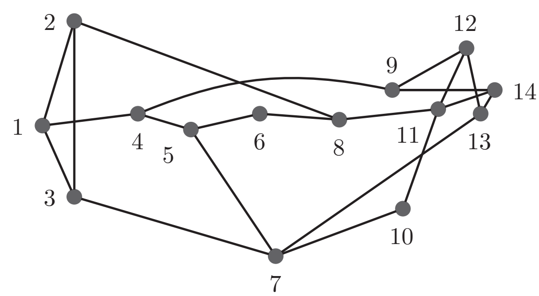

Fig. 2. 14 Node NSFNET backbone network.

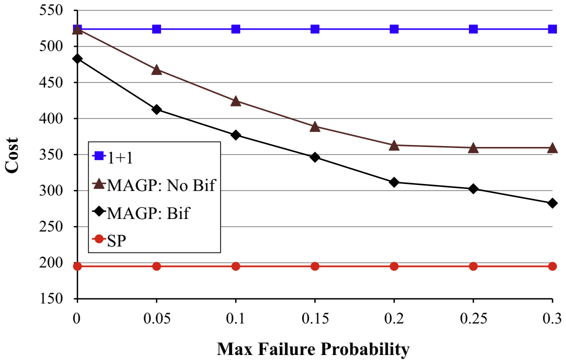

Fig. 3. Capacity cost vs. MFP with $q = \frac{1}{2}$.

ations is used as a lower bound. The cost of providing incremental protection with parameters q and P is the difference between the cost of the respective protection strategies and shortest path routing.

Note that allowing the primary flow to bifurcate allows requirements to be met using a lower cost allocation. This is because splitting the primary flow distributes the risk so that upon an edge failure, less primary flow is disrupted, which then requires less protection resources. If the flow is allowed to drop to $\frac{1}{2}$ for 1 out of 20 failures (5% of the time), then a savings of 24% in protection capacity is realized for the case with bifurcation, and 17% without bifurcation as compared to $1 + 1$ protection. As the flow is allowed to drop more often to its partial protection requirement after a failure, savings increase. For $P = 0.1$, a savings of 45% and 30% is seen for MAGP with and without bifurcation, respectively. For $P = 0.2$, the savings are 65% and 49%. Further increases in P result in only small additional savings; hence, the simulations were stopped at $P = 0.3$.

4. Optimal solution and algorithms without backup capacity sharing

While the MILP presented in the previous section finds an optimal solution to the multiple availability guaranteed protection problem, it is not a computationally efficient method of finding a solution, nor does it provide insight into why a solution is optimal. In this section, we analyze the MAGP problem to help develop efficient algorithms and heuristics for finding a minimum-cost routing when backup capacity sharing in the network is not allowed. The MAGP problem requires identifying a primary path such that segments of it are protected in a way that after a link failure, the flow drops to q with probability of at most P.

The case of $q = 0$ is explored in Section 4.1. When $q = 0$, there is no partial protection requirement, so there is only a single availability guarantee. This is the traditional availability guarantee, which has been examined in previous works. An optimal pseudo-polynomial algorithm is presented to solve MAGP with $q = 0$, which to the best of our knowledge is the first such algorithm. In Section 4.2, the case of $q > 0$ is examined. We show that finding a feasible solution to the closely related problem of singly constrained shortest pair of disjoint paths is strongly NP-Hard (there exists no pseudo-polynomial or ϵ-approximation algorithm), and conjecture that the MAGP problem with $q > 0$ is also strongly NP-Hard. Hence, a heuristic for solving MAGP with $q > 0$ is developed. Multiple availability guaranteed protection with the use of backup capacity sharing is examined in Section 5.

4.1. Availability guarantees with $q = 0$

When $q = 0$, the partial protection requirement is removed and no flow is needed during the downtime. To solve this problem, a primary path needs to be found such that segments of it are protected, and after a failure, the flow can drop to 0 with probability of at most P. First, a restricted version of the problem is considered where we try to meet availability requirements without the use of spare allocation. It can be shown that the solution to the restricted problem is the constrained shortest path (CSP) problem [23]. Next, the problem without restrictions on spare allocation is studied. We transform this unrestricted problem to an instance of the restricted one, and use CSP to find an optimal pseudo-polynomial algorithm for MAGP when $q = 0$.

4.1.1. Availability guarantees without spare allocation

First, we consider finding the lowest-cost path between s and t that meets the availability guarantee without the use of spare allocation. In other words, we want to find the lowest-cost path such that the sum of all the failure probabilities in that path are less than P. This problem is recognized to be the constrained shortest path problem (CSP) [23], which is NP-Hard. A dynamic program exists that finds the minimum-cost solution to CSP in pseudo-polynomial time [24], with a running time of $O(n^2P)$, where n is the number of nodes in the network; the P factor is what makes this running time pseudo-polynomial. CSP assumes all inputs are integer, so instead of the failure probabilities being between 0 and 1, we multiply P and all p_{ij} values, $\forall\{i,j\} \in E$, by the smallest factor that makes all the values integer (all inputs are assumed to be rational). Thus, for the remainder of this section, P and p_{ij} are assumed to be integer.

In general, a path may not exist from the source to the destination that can meet the availability requirement. Furthermore, if one exists, it is not necessarily of lowest cost. We next examine augmenting the flow with spare allocation to find a minimum-cost solution that meets requirements.

4.1.2. Availability guarantees with spare allocation

We now examine allowing the use of spare allocation to protect segments of the primary path in a manner that ensures the entire end-to-end path meets availability guarantees. If a failure of a segment in the primary path does not cause a disruption in the end-to-end flow, then that segment is considered protected. A routing that meets guarantees will be a concatenation of protected and unprotected segments. Fig. 4 shows a sample solution for a unit demand between v_1 and v_6 with $P = 0.2$, which illustrates how the use of spare allocation enables meeting availability

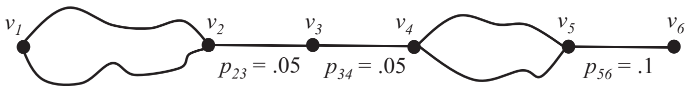

Fig. 4. Routing to meet $P = 0.2$ with $q = 0$ from v_1 to v_6.

guarantees. The probabilities of link failure are as labeled, and all lines represent a unit flow.

The primary segments between node pairs (v_2, v_4) and (v_5, v_6) are unprotected, and their total probability of failure must be at most the maximum failure probability of $P = 0.2$. The primary segments between node pairs (v_1, v_2) and (v_4, v_5) are completely protected with the primary path segment being protected by a disjoint backup path. After a failure of either of these protected primary segments, one unit of flow still remains; they contribute a total failure probability of zero to the routing. In this example, disjoint paths were used for the protected segments between node pairs (v_1, v_2) and (v_4, v_5). In fact, the lowest cost allocation to form a protected segment between any two nodes i and j is the minimum-cost pair of disjoint paths between the two, as demonstrated in Lemma 1.

Lemma 1. *The minimum-cost protected segment between nodes i and j is the minimum-cost pair of disjoint paths.*

Proof. For a segment between i and j to be protected, 1 unit of flow must remain between the source and destination after any single edge failure in the primary path. No backup edge will have an allocation greater than 1 because the primary flow will have 1 unit, and exactly 1 unit will need to be restored after any primary failure. An equivalent problem is to find the lowest-cost routing for 2 units of flow between i and j in a network where every edge has a maximum capacity of 1. After any single edge failure, at least 1 unit of flow will remain. This is a minimum-cost flow problem [23], whose solution has integer flows when given integer inputs. Since every edge has a capacity of 1, there will be two distinct edge-disjoint flows of 1 unit each. Clearly, the lowest-cost solution has these flows routed on the minimum-cost pair of disjoint paths. □

Using Lemma 1, every possible protected segment between any two nodes can be transformed to a single edge with a failure probability of 0 and a cost equivalent to the minimum-cost pair of disjoint paths between the nodes. We denote the cost and probability of the minimum-cost pair of disjoint paths between nodes i and j as $\hat{c}_{ij}$ and $\hat{p}_{ij} = 0$, respectively. Now, any protected segment between some node pair (i,j) can be represented as a single edge between i and j in the network. This edge contains the primary and spare allocation that would be used if a protected segment between i and j was needed. Adding an edge for every possible protected segment transforms the problem back to the restricted version where no spare allocation was allowed. This problem can now be solved using the constrained shortest path algorithm.

Our proposed algorithm is as follows. We take the graph G, where every edge has a cost and a failure probability associated with it, and augment the graph with an edge between every pair of nodes (i,j) such that the cost of that edge is the minimum-cost pair of disjoint paths between i and j, and the probability of failure for that edge is zero. Thus the new augmented graph has two kinds of edges between nodes i and j: unprotected edges corresponding to the original edges in graph G (if such an edge existed) with a cost c_{ij} and failure probability p_{ij}, and protected edges with a cost $\hat{c}_{ij}$ and failure probability $\hat{p}_{ij} = 0$, where $\hat{c}_{ij}$ is the cost of the shortest pair of disjoint paths between i and j. We next run the constrained shortest path algorithm on the augmented graph to find the minimum-cost solution. We call this algorithm the Segment Protected Availability Guaranteed Algorithm (SPAG).

Theorem 2. *SPAG will return a minimum-cost routing, if one exists, and has a running time of* $O(n^4 log(n) + n^2 P)$.

Proof. To meet availability requirements, a solution will have a primary path that consists of a combination of protected and unprotected segments. As shown in Lemma 1, a protected segment between any two nodes is the shortest pair of disjoint paths between those nodes. Using the above graph augmentation, an edge is added for every feasible protected segment. The constrained shortest path algorithm then evaluates every possible combination of protected and unprotected segments to find the lowest cost solution between the source and destination.

For the running time, the $O(n^4 log(n))$ component comes from $O(n^2)$ iterations of the shortest pair of disjoint paths algorithm (there are $O(n^2)$ node pairs), which takes $O(n^2 log(n))$ time per iteration [22]. The recursion for the constrained shortest path problem runs in $O(n^2 P)$ time. □

A simulation similar to that of Section 3.2 was used to compare SPAG to the optimal solution without bifurcation for $q = 0$. Simulation results show that SPAG is in fact optimal for all tested demands.

4.2. Meeting availability requirements with $q > 0$

Next, we examine the case of $q > 0$. The problem now has multiple availability guarantees: after an edge failure in the primary path, the flow either remains at 1 or, with at most a probability of P, drops to q. Consider a sample solution shown for a unit demand between nodes v_1 and v_6 with a maximum failure probability of 0.2 in Fig. 5, which consists of alternating fully-protected and q-protected segments (the dotted line being the q flow). Between node pairs (v_1, v_2) and (v_4, v_5), the primary segments are fully protected, and a failure in those primary segments will not cause a drop in end-to-end flow. Between node pairs (v_2, v_4) and (v_5, v_6), the primary segments have q flow routed on a segment that is edge-disjoint; after a failure in the

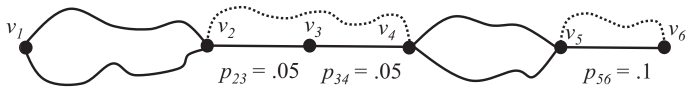

Fig. 5. Routing to meet $P = 0.2$ and $q > 0$ from v_1 to v_6.

primary segment, flow will drop to q with a probability of at most 0.2. We conjecture that the multiple availability guaranteed protection problem with $q > 0$ is *strongly* NP-Hard[1] by demonstrating the complexity of a related disjoint paths problem, previously unexplored in the literature, to be strongly NP-Hard. Using this result, we present an efficient heuristic for solving the $q > 0$ case.

A sample solution is shown in Fig. 5, which consists of alternating fully-protected and q-protected segments (the dotted line being the q flow).

We consider protecting a q-protected segment by finding a pair of disjoint paths between i and j such that one of them is constrained: the primary segment is constrained to have a probability of failure of at most P. We call this problem the Singly Constrained Shortest Pair of Disjoint Paths (SCSPD). There has been work trying to find the shortest pair of disjoint paths such that each path is constrained by the same parameter [25]. The authors of [25] found that this doubly constrained problem, while NP-Hard, has an ϵ-approximation algorithm. Their problem is distinct from ours in that SCSPD only constrains one of the two paths. Clearly, a solution to the doubly constrained problem is a feasible solution to the singly constrained one, but it is not necessarily optimal, and a lack of a solution to the former does not imply the non-existence of a solution to the latter. In fact, we show that when the constraint is relaxed for one of the paths, SCSPD becomes *strongly* NP-Hard, which means that a solution cannot be ϵ-approximated, nor can there be any pseudo-polynomial algorithm for optimality [21].

Theorem 3. *The singly constrained shortest pair of disjoint paths problem is strongly NP-Hard.*

Proof. To demonstrate strong NP-Hardness of SCSPD, a reduction from the 3SAT problem [26] is performed. See Appendix B for the complete proof. □

Since SCSPD is strongly NP-Hard, the dynamic programming approach used to solve for $q = 0$ does not work when $q > 0$. We conjecture that the multiple availability guaranteed protection when $q > 0$ is in fact also strongly NP-Hard, thereby necessitating a heuristic approach to solve the problem. Our proposed heuristic augments the $q = 0$ algorithm: after an optimal solution for $q = 0$ is found, find the shortest disjoint path for the unprotected segments and allocate a flow of q to them. We call this algorithm the Segment Protected Multiple Availability Guaranteed Algorithm (SPMAG).

A simulation similar to that of Section 3.2 was run comparing SPMAG and the optimal solution to MAGP without

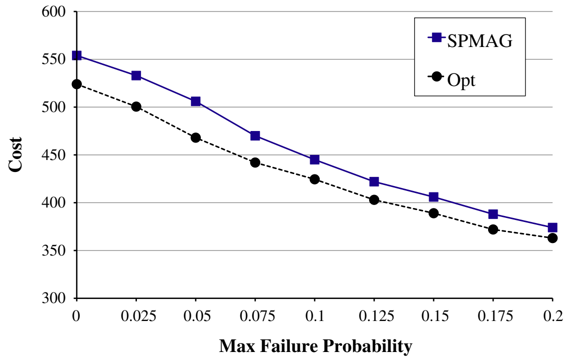

Fig. 6. SPMAG capacity cost vs. MFP with $q = \frac{1}{2}$.

flow bifurcation; the results are plotted in Fig. 6. On average, SPMAG performs within 6% of the optimal solution to the multiple availability guaranteed protection problem.

5. Algorithm with backup capacity sharing

In the previous section, efficient algorithms were presented without the use of backup capacity sharing, including an optimal algorithm for the case of $q = 0$, which corresponds to the standard availability constraint. These results are useful for a basic understanding of the Multiple Availability Guaranteed Protection problem (MAGP), and for networks that do not allow protection sharing. But many times, networks do utilize backup sharing, and significant savings can often be achieved. In this section, a time-efficient algorithm for MAGP using backup capacity sharing is presented.

If two primary flows for two different demands are edge-disjoint from one another, then under a single-link failure model, at most one can be disrupted at any given point in time. Since at most one demand will need to be restored after a failure, two failure-disjoint flows can share backup capacity.

Determining how much backup capacity can be shared for guaranteed path protection was examined in [2,3]. These papers use conflict sets to determine potential backup sharing on an edge by keeping track of how much backup capacity was allocated on one edge to protect against the failure of another. If more backup capacity is already allocated on some edge than is needed to protect for the failure of another edge, then that edge's backup capacity can be shared. This model can be extended to the partial protection framework by guaranteeing that any particular demand has its partial flow requirement met after a failure. An example without probabilistic availability guarantees is given in Fig. 7. Define the variable h_{ij}^{kl} to be the number of units of capacity used on edge $\{i,j\}$ to protect against the failure of edge $\{k,l\}$. The maximum

[1] In addition to being NP-Hard, a problem that is Strongly NP-Hard indicates that there exists no pseudo-polynomial or ϵ-approximation algorithm for finding a solution [21].

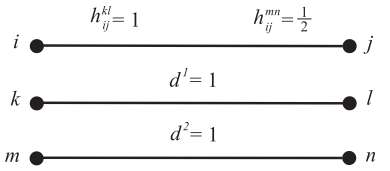

Fig. 7. Example of a conflict set with partial protection.

number of units allocated on edge $\{i,j\}$ to protect against any edge failure is the total spare allocation needed on $\{i,j\}$. In Fig. 7, two demands with $q^1 = 1$ and $q^2 = \frac{1}{2}$ are routed. Both demands use edge $\{i,j\}$ for protection with 1 unit being needed after the failure of $\{k,l\}$ and $\frac{1}{2}$ unit being needed after the failure of $\{m,n\}$. In this example we have $h_{ij}^{kl} = 1$ and $h_{ij}^{mn} = \frac{1}{2}$.

Now, consider a new demand with $q^3 = \frac{1}{2}$. If this demand were to have its primary flow routed on edge $\{k,l\}$ and use $\{i,j\}$ for protection, h_{ij}^{kl} will increase by $\frac{1}{2}$ unit. Since the amount of spare allocation on an edge is the maximum capacity needed to protect against any edge failure, the total allocation will increase by $\frac{1}{2}$. Alternatively, if the demand were to use $\{m,n\}$ instead of $\{k,l\}$, h_{ij}^{mn} will increase by $\frac{1}{2}$, and the maximum number of units needed to protect against any edge failure will still only be 1. No additional resources are required for protection on $\{i,j\}$ under this routing scenario. For the exact implementation of conflict sets for protection resource sharing, see [2,3].

We now consider meeting probabilistic availability guarantees. Given some primary path between s and t, certain segments will be fully-protected, and others will be partially protected with a flow of q. For each edge in the primary path, the cost of using 1 or q units on edge $\{i,j\}$ for backup is calculated using conflict sets. For a primary path with a set of edges S, let $\beta(S,1)$ be the cost of backup edges for fully protecting any edge, and $\beta(S,q)$ be the cost of backup edges that partially protect an edge with a flow of q.

Next, we calculate the cost of protecting each possible segment of a given primary path with either full or partial protection; if there are v nodes in the primary path, then there are $\frac{v(v-1)}{2}$ segments contained within that path. We construct a new graph G_S^{st} with two edges between every pair of nodes in the primary path; these two edges correspond to fully or partially protecting the primary segment between nodes i and j. For every primary segment in the primary path, we find two paths that are disjoint to that segment: one that fully protects that segment, and one that partially protects it. For full protection, the edge between nodes i and j in G_S^{st} is the shortest disjoint path to primary segment (i,j) using the set of edge costs $\beta(S,1)$; for partial protection, the edge between i and j in G_S^{st} is the shortest disjoint path to primary segment (i,j) using $\beta(S,q)$. The cost of the edge in G_S^{st} to fully protect primary segment (i,j) is c_{ij}^1, and has failure probability $p_{ij}^1 = 0$. The cost of the edge in G_S^{st} to partially protect primary segment (i,j) is c_{ij}^q, and has failure probability p_{ij}^q equal to the failure probability of the primary path segment between nodes i and j. Once G_S^{st} is fully constructed, we find the constrained shortest path in G_S^{st} from s to t with a maximum failure probability of P^{st}. This path will be the backup, which meets all partial protection and availability requirements when combined with the initial primary path.

Since jointly optimizing the primary and protection path with backup sharing is NP-Hard [2], we choose a simple strategy of using the shortest path for the primary. This is in contrast to [2,3] that offer a heuristic approach to jointly optimize the primary and protection path for each incoming demand. Our simulations (presented at the end of this section) show that using the shortest path for the primary route actually performs better than jointly optimizing the primary and backup paths for each incoming demand.

The algorithm for MAGP to route a demand using backup capacity sharing is as follows. For an arriving demand between s and t, find the shortest path between those two nodes; S^{st} will be the set of edges in that shortest path. We then construct the graph G_S^{st} using the procedure discussed above, and then we find the lowest-cost shared backup to meet protection and availability requirements. This algorithm is called Dynamic Multiple Availability Guaranteed Segment Protection (DMAGSP). The pseudocode for this algorithm can be found in Appendix C.

An example is shown in Fig. 8. A unit demand needs to be routed between v_1 and v_4, with a maximum failure probability of $P = 0.2$. For this example, the shortest path between v_1 and v_4 has already been found and is given, with this path being $v_1 - v_2 - v_3 - v_4$. This shortest path is chosen to be the primary route; failure probabilities for the edges of that primary path are as labeled in the figure.

For demonstration purposes, the protection paths protecting each possible segment of the primary path are already computed, and are shown in Fig. 8a. For each segment of the primary path, two arcs are constructed between that segment's two end nodes: one that fully protects against a failure in that segment, having probability of failure $p_{ij}^1 = 0$, and one that q-protects that segment,

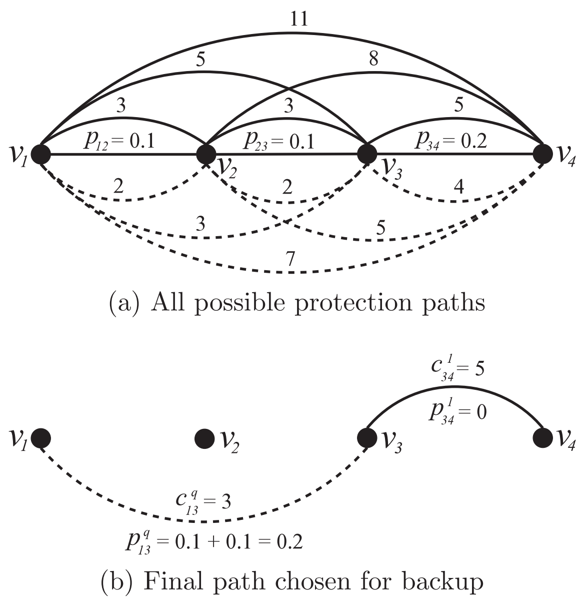

Fig. 8. Example of algorithm with $P = 0.2$.

with probability of failure p_{ij}^{q} equal to the sum of the edges' failure probabilities in that primary segment. The arcs above the primary path are the lowest-cost full protection paths for each segment of the primary, and the arcs below the path are the lowest-cost partial protection paths. Costs of protecting the primary path segments with 1 or q units of flow are labeled on the arcs.

The protection paths for each of the primary path segments (the arcs above and below the primary path), form the new graph G_S^{st}. The constrained shortest path algorithm is run on G_S^{st} between v_1 and v_4 with a maximum failure probability of P, which returns the final backup path. The backup path for this example with $P = 0.2$, as shown in Fig. 8b, meets all protection and availability requirements when combined with the primary path found previously. In this example, the segment between v_1 and v_3 is q protected, whereas the segment between v_3 and v_4 is fully protected.

A simulation similar to that of Section 3.2 was run, this time with demands arriving dynamically at random and serviced one-at-a-time in the order of their arrival. The protection requirement q for each demand is drawn from a truncated normal distribution with mean of $q = \frac{1}{2}$ and standard deviation $\sigma = \frac{1}{2}$. The maximum failure probability P has a truncated normal distribution with a standard deviation $\sigma = .025$; the mean of P is varied between 0 and 0.3. We compare multiple availability guaranteed protection with and without sharing (which jointly optimizes the primary and backup path for each incoming demand), DMAGSP, and 1:1 protection with backup capacity sharing.

The capacity needed to route the demand and protection flows are plotted in Fig. 9 as a function of the expected value of P. Again, the shortest path routing without protection considerations is used as a lower bound for the capacity allocation. MAGP with backup sharing, which jointly optimizes the primary and backup paths for each incoming demand, achieves an average reduction in excess resources of 42% over 1:1 protection with sharing for all values of P that were tested, and an average reduction of 51% over MAGP without sharing.

A notable result is that Dynamic Multiple Availability Guaranteed Segment Protection (DMAGSP) in fact performs *better* than the greedy optimal solution with dynamic arrivals. This can be explained by observing that the algorithm takes the simple strategy of the shortest path as the primary for each connection, as opposed to jointly optimizing the primary and backup routes, which may take a longer primary path to take advantage of backup sharing. This longer path makes it potentially more difficult for future demands to find disjoint primary routes, lowering their ability to share protection resources. While other works have focused on finding heuristics to jointly optimize the primary and backup paths for each incoming demand, it appears a better approach is to simply take the shortest path for the primary route.

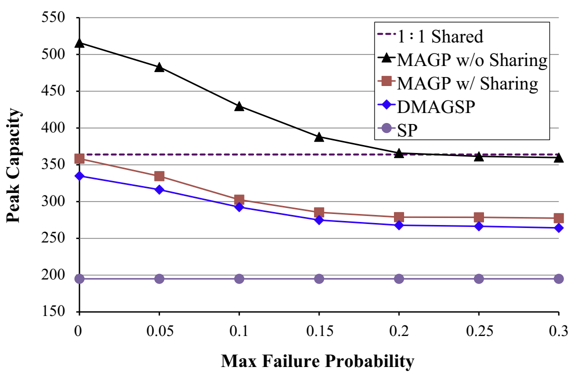

Fig. 9. Peak capacity cost vs. MFP with $q = \frac{1}{2}$.

6. Conclusion

In this paper, a novel network protection scheme with multiple availability guarantees was introduced. In particular, the multiple availability guarantees will maintain the full demand for at least a guaranteed fraction of time and guarantee a partial flow during the downtime. If the demand is allowed to drop to 50% of its flow for only 1 out of every 20 failures, a 24% reduction in excess resources can be realized over the traditional disjoint path full protection schemes. For the $q = 0$, which corresponds to the previously studied scenario where full availability is guaranteed for a fraction of time, we developed an optimal pseudo-polynomial algorithm. For the case of $q > 0$, we developed a time-efficient heuristic (segment protected multiple availability guaranteed protection) that performs within 6% of the optimal solution to the multiple availability guaranteed protection problem. We then extended the Multiple Availability Guaranteed Problem (MAGP) to the case where backup capacity sharing is utilized to lower the total amount of resources needed to meet protection and availability requirements. An algorithm for MAGP with protection sharing was developed for dynamic arrivals, which in fact performs better than jointly optimizing the primary and backup paths for each incoming demand.

Appendix A. Proof of NP-Hardness for multiple availability guaranteed protection

To demonstrate NP-Hardness of MAGP, the 1–0 knapsack problem [21] will be reduced to MAGP. The knapsack problem finds the maximum value subset of k items, with the i^{th} item having cost c_i and weight p_i, such that the maximum weight P of the knapsack is not exceeded.

Consider the network shown in Fig. A.10 with link costs and probabilities denoted by c_i and p_i, respectively. We wish to find a minimum-cost routing for a unit demand from s to t with a maximum failure probability P and partial protection requirement $q = 0$. After any link failure, the network will either maintain its full flow of 1 unit, or have no flow with a probability of at most P. There are k distinct link groups, where each of the two links in any group have the same probability of failure and cost. Primary flow has to be allocated onto at least one of these links, otherwise the primary demand cannot be met. If the network maintains full connectivity after a primary failure in the kth link

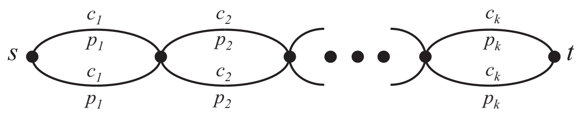

Fig. A.10. Sample network for MAGP NP-Hardness proof.

group, then each link in that group will have an allocation of 1 unit. If there is no flow after a link failure, then only one link has an allocation of 1, and the other 0. So, every link group has at least one link with a flow of 1, which is a fixed cost regardless of protection allocation.

To find the lowest cost protection allocation to meet availability guarantees, we need to find the lowest cost combination of the remaining links after the primary flow is allocated such that the sum of the failure probabilities for the links that have no allocation are less than P. Our objective is $\min\sum_{i=1}^{k} c_i(1-z_i)$, subject to the constraint of $\sum_{i=1}^{k} p_i z_i \leqslant P$, with $z_i \in \{0,1\}\ \forall i \in 1,\ldots,k$. The objective can be rewritten to maximize the cost of the links that do not have allocation: $\min\sum_{i=1}^{k} c_i(1-z_i) = \max\sum_{i=1}^{k} c_i z_i$. We now recognize this to be the NP-Hard 1–0 knapsack problem with a maximum weight of P, and cost and weight of the i^{th} item being the cost and probability, respectively, of each pair of links in the ith link group. If there existed a polynomial time solution to MAGP, then there would exist one for the 1–0 knapsack problem. Therefore, MAGP is at least as hard as the 1–0 knapsack problem.

Appendix B. Proof of strong NP-Hardness for singly constrained shortest pair of disjoint paths

To prove that SCSPDP is strongly NP-Hard, we borrow a reduction that demonstrates the NP-Hardness of a different, but similar, problem [26] and adapt it to the SCSPD problem. The authors of [26] attempt to find the "min-min" disjoint pair of paths, which is defined as the minimum-cost pair of disjoint paths that contains, over all sets of possible disjoint paths, the minimum-cost shorter path.

To demonstrate this problem is NP-Hard, they construct a mapping of the 3*SAT* problem to a graph where a solution to their problem will simultaneously solve the 3*SAT* problem. A solution to 3*SAT* determines if there exists a 1/0 assignment to the variables that will make a specific boolean expression true [21]. The graph in Fig. B.11 is a sample network corresponding to the instance of the 3*SAT* problem of $(x_1 \vee x_2 \vee x_3) \wedge (\hat{x}_1 \vee \hat{x}_3 \vee x_4) \wedge (x_2 \vee \hat{x}_3 \vee x_4) \wedge (\hat{x}_2 \vee x_3 \vee \hat{x}_4)$ [26]. Without going into the specifics of the reduction (see [26] for details), a generalized version of their result is: if two disjoint paths can be found between s and t such that one of them uses only the dotted lines, then that solution is also a solution to the 3*SAT* problem (see [26] for the proof). To demonstrate strong NP-Hardness, one needs to show the problem remains NP-Hard after the value of all inputs to the system have been bounded by some polynomial [21].

We will first show the problem to be NP-Hard by assigning costs and probabilities to the edges of the above network such that solving SCSPD will also solve the 3*SAT* problem. Then, we will demonstrate that SCSPD

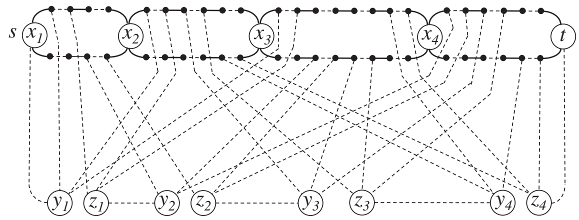

Fig. B.11. Sample network to solve an instance of 3*SAT* from [26].

is in fact strongly NP-Hard. Assume there exists D dotted edges and L solid edges in the 3*SAT* reduced graph. Since we can assign parameters of our choosing to the edges, assign a cost of 0 for the dotted edges and a cost of 1 to the solid edges. We choose the failure probability of each dotted edge to be $\frac{\alpha}{D}$ and the probability of each solid edge to be $\frac{1-\alpha}{L}$, such that $\alpha \leqslant \frac{1-\alpha}{L} \to \alpha \leqslant \frac{1}{1+L}$. Additionally, choose a maximum probability of failure P such that $\alpha \leqslant P < \frac{1-\alpha}{L}$. Since using any solid edge will make that path violate the maximum failure probability P, the only feasible solution to SCSPD on this network is for the constrained path to use only dotted edges. But if such a solution could be found, it would solve the 3*SAT* problem, which is NP-Hard. The problem in fact remains NP-hard when all input values are polynomial bounded: L and D are polynomial bounded by the number of inputs from the 3*SAT* problem, and α can be chosen to be polynomial bounded. If all input parameters to a problem are bounded by some polynomial, and the problem remains NP-Hard, then the problem is *strongly* NP-Hard [21]. Finding any feasible solution to SCSPD on this bounded graph will still solve the 3SAT problem. Therefore, SCSPD is strongly NP-Hard.

Appendix C. Pseudocode for DMAGSP

The following are input parameters to Dynamic Multiple Availability Guaranteed Segment Protection (DMAGSP) algorithm. Graph $G = (V, E, \mathcal{P}, C) : E$ is the set of edges; V is the set of vertices; $\mathcal{P}$ are the set of edge failure probabilities; C are the edge costs. Λ is the set of paths for the primary flows already allocated, with λ^{ab} being the set of edge allocations for the primary flow for the demand between nodes a and b. Δ is the set of backup flow allocations, with δ_{ij}^{ab} being the amount of protection flow allocated to edge $\{i,j\}$ to protect against a failure of the primary path for the demand between nodes (a,b). The current demand to be routed is from node s to t, with a demand of d, partial protection requirement of q, and maximum failure probability of P. The output of the DMAGSP algorithm will be a primary path from s to t using the set of edges S^{st} that carries a flow of d, and a backup path using the set of edges B^{st}, with allocation b_{ij}^{st} on edge $\{i,j\}$ of that path. We first present the pseudocode for the function shared_backup in Algorithm 1, which will find the lowest cost path, carrying a flow of f, that is disjoint to the primary path between u and v, which has a set of edges S. The output of the function will be the set of edges for the backup path B, and the cost of that backup path β. We then present the pseudocode for the full Dynamic Multiple Availability Guaranteed Segment Protection (DMAGSP) scheme in Algorithm 2. We refer the reader to Fig. 8 from Section 5 for an illustrative example.

Algorithm 1. (B, β) = shared_backup$(G, \Lambda, \Delta, S, u, v, f)$.

1: Determine the set of primary paths from Λ that are edge-disjoint to S; label this set of primary paths as Γ
2: **for** $\forall\{i,j\} \in E \setminus S$ **do**
3: $c^b_{ij} = \max\left(c_{ij}(f - \sum_{(a,b)\in\Gamma}\delta^{ab}_{ij}),\ 0\right)$ *// On edge {i,j}, set the cost of protection capacity that can be shared to zero*
4: **end for**
5: The edge costs c^b_{ij} form the set C^b
6: Using the set of edges $E \setminus S$, with edge costs C^b, find the shortest path from u to v
7: Label the set of edges of this path B, and set $\beta = \sum_{\{i,j\}\in B} c^b_{ij}$

Algorithm 2. (S^{st}, B^{st}) = DMAGSP$(G, \Lambda, \Delta, s, t, d, q, P)$.

1: Find a shortest path from s to t; label these set of edges as S^{st}
2: **for** every possible path segment S^{uv} of S^{st} (the path segment goes from node u to v) **do**
3: Find the shared protection path for the full demand d and the partial protection requirement qd:
$\left(B(S^{uv}, 1), \beta(S^{uv}, 1)\right)$ = shared_backup$(G, \Lambda, \Delta, S^{uv}, u, v, d)$
$\left(B(S^{uv}, q), \beta(S^{uv}, q)\right)$ = shared_backup$(G, \Lambda, \Delta, S^{uv}, u, v, dq)$
4: In protection graph G^{st}_S, create two edges between nodes u and v:
The first edge has a cost $\beta(S^{uv}, 1)$, and probability of failure $p^1_{uv} = 0$
The second having a cost $\beta(S^{uv}, q)$, and probability of failure $p^q_{uv} = \sum_{\{i,j\}\in S^{uv}} p_{ij}$
5: **end for**
6: In protection graph G^{st}_S, find the constrained shortest path [24] from node s to t such that the sum of the failure probabilities for the edges of that path do not exceed P
7: Label the set of edges of this path as B^{st}

References

[1] S. Ramamurthy, L. Sahasrabuddhe, B. Mukherjee, Survivable WDM mesh networks, J. Lightwave Technol. 21 (2003) 870–884.
[2] C. Ou, J. Zhang, H. Zang, L. Sahasrabuddhe, B. Mukherjee, New and improved approaches for shared-path protection in WDM mesh networks, J. Lightwave Technol. 22 (2004) 1223–1232.
[3] G. Mohan, S. Murthy, A. Somani, Efficient algorithms for routing dependable connections in WDM optical networks, IEEE/ACM Trans. Network. 9 (2002) 553–566.
[4] B. Mukherjee, WDM optical communication networks: progress and challenges, IEEE J. Select. Areas Commun. 18 (2000) 1810–1824.
[5] W. Yao, B. Ramamurthy, Survivable traffic grooming with path protection at the connection level in WDM mesh networks, J. Lightwave Technol. 23 (2005) 2846.
[6] H. Wang, E. Modiano, M. Médard, Partial path protection for WDM networks: end-to-end recovery using local failure information, in: Proceedings of Seventh International Symposium on Computers and Communications, ISCC 2002, IEEE, 2002, pp. 719–725.
[7] W. Grover, Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking, Prentice Hall, 2004.
[8] A. Saleh, J. Simmons, Evolution toward the next-generation core optical network, J. Lightwave Technol. 24 (2006) 3303–3321.
[9] J. Zhang, K. Zhu, H. Zang, N. Matloff, B. Mukherjee, Availability-aware provisioning strategies for differentiated protection services in wavelength-convertible WDM mesh networks, IEEE/ACM Trans. Network. (TON) 15 (2007) 1177–1190.
[10] C. Saradhi, M. Gurusamy, L. Zhou, Differentiated QOS for survivable WDM optical networks, Commun. Magaz., IEEE 42 (2004) S8–14.
[11] M. Tacca, A. Fumagalli, A. Paradisi, F. Unghváry, K. Gadhiraju, S. Lakshmanan, S. Rossi, A. Sachs, D. Shah, Differentiated reliability in optical networks: theoretical and practical results, J. Lightwave Technol. 21 (2003) 2576–2586.
[12] R. Banner, A. Orda, The power of tuning: a novel approach for the efficient design of survivable networks, IEEE/ACM Trans. Network. (TON) 15 (2007) 737–749.
[13] L. Song, J. Zhang, B. Mukherjee, Dynamic provisioning with availability guarantee for differentiated services in survivable mesh networks, IEEE J. Select. Areas Commun. (JSAC) 25 (2007) 35–43.
[14] W. Yao, B. Ramamurthy, Survivable traffic grooming with differentiated end-to-end availability guarantees in WDM mesh networks, in: The 13th IEEE Workshop on Local and Metropolitan Area Networks, 2004. LANMAN 2004, IEEE, 2004, pp. 87–90.
[15] J. Zhang, K. Zhu, H. Zang, B. Mukherjee, Service provisioning to provide per-connection-based availability guarantee in WDM mesh networks, in: Optical Fiber Communication Conference, Optical Society of America, 2003, pp. 622–624.
[16] S. Rai, O. Deshpande, C. Ou, C. Martel, B. Mukherjee, Reliable multipath provisioning for high-capacity backbone mesh networks, IEEE/ACM Trans. Network. (TON) 15 (2007) 803–812.
[17] O. Gerstel, G. Sasaki, Quality of protection (QoP): a quantitative unifying paradigm to protection service grades, Opt. Netw. Magaz. 3 (2002) 40–49.
[18] G. Kuperman, E. Modiano, A. Narula-Tam, Analysis and algorithms for partial protection in mesh networks, in: INFOCOM, 2011 Proceedings IEEE, IEEE, 2011, pp. 516–520.
[19] G. Kuperman, E. Modiano, A. Narula-Tam, Partial protection in networks with backup capacity sharing, in: National Fiber Optic Engineers Conference, Optical Society of America, 2012, pp. 1–3.
[20] G. Kuperman, E. Modiano, A. Narula-Tam, Network protection with multiple availability guarantees, in: 2012 IEEE International Conference on Communications (ICC), IEEE, 2012, pp. 6241–6246.
[21] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, CA, 1979.
[22] J.W. Suurballe, R.E. Tarjan, A quick method for finding shortest pairs of disjoint paths, Networks 14 (1984) 325–336.
[23] R. Ahuja, T. Magnanti, J. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice-Hall, New Jersey, 1993.
[24] H. Joksch, The shortest route problem with constraints, J. Math. Anal. Appl. 14 (1966) 191–197.
[25] A. Orda, A. Sprintson, Efficient algorithms for computing disjoint QOS paths, Twenty-Third Annual Conference of the IEEE Computer and Communications Societies, INFOCOM 2004, vol. 1, IEEE, 2003, pp. 727–738.
[26] D. Xu, Y. Chen, Y. Xiong, C. Qiao, X. He, On finding disjoint paths in single and dual link cost networks, Twenty-Third Annual Conference of the IEEE Computer and Communications Societies, INFOCOM 2004, vol. 1, IEEE, 2004, pp. 705–715.

Greg Kuperman received his Ph.D. in Communications and Networking from the Massachusetts Institute of Technology in 2013, and he received his B.S.E in Computer and Telecommunication Engineering and M.S.E. in Electrical Engineering from the University of Pennsylvania in 2005. He is currently a postdoctoral researcher at MIT in the Laboratory of Information and Decision Systems (LIDS).

Eytan Modiano received his B.S. degree in Electrical Engineering and Computer Science from the University of Connecticut at Storrs in 1986 and his M.S. and Ph.D. degrees, both in Electrical Engineering, from the University of Maryland, College Park, MD, in 1989 and 1992 respectively. He was a Naval Research Laboratory Fellow between 1987 and 1992 and a National Research Council Post Doctoral Fellow during 1992–1993. Between 1993 and 1999 he was with MIT Lincoln Laboratory where he was a project leader for MIT Lincoln Laboratory's Next Generation Internet (NGI) project. Since 1999 he has been on the faculty at MIT, where he is a Professor in the Department of Aeronautics and Astronautics and the Laboratory for Information and Decision Systems (LIDS). His research is on communication networks and protocols with emphasis on satellite, wireless, and optical networks. He is currently an Associate Editor for IEEE/ACM Transactions on Networking. He had served as Associate Editor for IEEE Transactions on Information Theory, and as guest editor for IEEE JSAC special issue on WDM network architectures; the Computer Networks Journal special issue on Broadband Internet Access; the Journal of Communications and Networks special issue on Wireless Ad-Hoc Networks; and for IEEE Journal of Lightwave Technology special issue on Optical Networks. He was the Technical Program co-chair for IEEE Wiopt 2006, IEEE Infocom 2007, and ACM MobiHoc 2007. He is a Fellow of the IEEE and an Associate Fellow of the AIAA.

Aradhana Narula-Tam is an Assistant Leader of the Airborne Networks Group at MIT Lincoln Laboratory. She joined the laboratory in 1998 and has worked in a variety of areas including optical networking, dynamic resource allocation for satellite communication systems, and IP Quality of Service. She was the Principal Investigator for MIT Lincoln Laboratory's efforts on the DARPA Core Optical Networking program prior to taking a temporary position as an Air Force civilian to lead the Modem Processor Group product team for the Family of Advanced Beyond Line-of-Sight Terminal (FAB-T) program from 2009 to 2010. Upon returning to the laboratory, she has focused on research and design of Line-of-Sight communication technologies and network architectures for highly mobile platforms in tactical environments. Dr. Narula-Tam holds a BSE degree from the University of Pennsylvania and SM and Ph.D. degrees from the Massachusetts Institute of Technology, all in electrical engineering. Her doctoral thesis analyzed theoretical performance limits of antenna diversity methods for mobile communications. She has served as session and track chairs for IEEE MILCOM and co-authored over 30 papers in the areas of wireless communications, satellite communication systems and networking, and optical networking.