Availability-based path selection

Yang, S; Trajanovski, S; Kuipers, Fernando

DOI
https://doi.org/10.1109/RNDM.2014.7014929

Publication date
2014

Document Version
Accepted author manuscript

Published in
Proceedings of the 6th International Workshop on Reliable Networks Design and Modeling

Availability-Based Path Selection

Song Yang, Stojan Trajanovski and Fernando A. Kuipers
Delft University of Technology, The Netherlands
{S.Yang, S.Trajanovski, F.A.Kuipers}@tudelft.nl

***Abstract*—In data communication networks, connection availability, which is defined as the probability that the corresponding connection will be found in the operating state, is a key element of many Service Level Agreements (SLA). The path over which a connection is to be established should obey the agreed-upon availability, otherwise the service provider may face revenue loss as stipulated in the SLA.**

In this paper, we study the problem of establishing a connection over at most k (partially) link-disjoint paths for which the availability is no less than δ ($0 < \delta \leq 1$). We consider networks with and without Shared-Risk Link Groups (SRLGs). We prove that this problem, in general, cannot be approximated in polynomial time, unless P=NP. We subsequently propose a polynomial-time heuristic algorithm and an exact Integer Nonlinear Programming (INLP) formulation for availability-based path selection. Finally, the proposed algorithms and two existing heuristic algorithms are compared in terms of acceptance ratio and running time.

***Index Terms*—Availability, Routing, Survivability, SRLG.**

I. Introduction

Due to the importance of data communication networks, even short service disruptions may result in significant economic loss. Hence, survivability mechanisms to protect connections are called for. For instance, by allocating a pair of link-disjoint paths (instead of only one unprotected path), data is transported by the primary path, and upon link failure, can be switched to the backup path.

Ideally, a survivability mechanism should also take into account the reliability of links. For instance, if both primary and backup paths contain links that have a high probability to be unavailable, then proper protection cannot be provided. Connection availability, a value between 0 and 1, is therefore important and refers to the probability that a connection (including its survivability mechanism) is in the operating state during the requested life-time of the connection.

However, a survivability mechanism that does not allow for more than 2 link-disjoint paths for each connection may still fail to satisfy the customer's availability requirement and $k > 2$ link-disjoint paths may be needed. Obviously, the bigger k is, the greater the availability of the connection could be, but also the greater the resource consumption (e.g., bandwidth) and hence price. This paper deals with the following Availability-Based Path Selection (ABPS) problem:

Definition 1: Given a network represented by $G(\mathcal{N}, \mathcal{L})$ where $\mathcal{N}$ represents the set of N nodes and $\mathcal{L}$ denotes the set of L links, and each link l has its own availability value A_l. For a request represented by $r(s, t, \delta)$, where s and t denote the source and destination, and δ $(0 < \delta \leq 1)$ represents the availability requirement, establish a connection over at most k (partially) link-disjoint paths for which the availability is at least δ.

Our key contributions are as follows:

1) We consider the ABPS problem in networks with and without Shared-Risk Link Groups (SRLGs).
2) We prove that, in general, the ABPS problem cannot be approximated in polynomial time.
3) We propose a polynomial-time heuristic algorithm and an exact Integer Nonlinear Programming (INLP) formulation to solve the ABPS problem.
4) We compare, via simulations, the proposed algorithms with two existing algorithms in terms of performance and running time.

The remainder of this paper is organized as follows. Section II explains the calculation of availability for different path types (unprotected path, k fully link-disjoint and k partially link-disjoint). In Section III, we analyze the complexity of the considered Availability-Based Path Selection (ABPS) problem. In Section IV, we consider the ABPS problem in SRLG networks. Section V presents our heuristic routing algorithm and an exact INLP formulation. Section VI provides our simulation results and an overview of the related work is presented in Section VII. We conclude in Section VIII.

II. Connection Availability

The availability of a system is the fraction of time the system is operational during the entire service time. Like the papers mentioned in the related work section, we assume that the link availabilities are uncorrelated/independent. If a connection is carried by a single (unprotected) path, its availability is equal to the path availability; if it is protected by $k \geq 2$ disjoint paths, the availability will be determined by these k protection paths. The availability A_j of a network component j can be calculated as [1]:

$$A_j = \frac{MTTF}{MTTF + MTTR} \tag{1}$$

where $MTTF$ represents Mean Time To Failure and $MTTR$ denotes Mean Time To Repair.

A. End-to-End Path Availability

We assume that the link availability is equal to the product of availabilities of all its components (e.g., amplifiers). We also assume multiple link failures may occur. Hence, if a path contains the links l_1, l_2, l_3,..., l_m, and their corresponding

availabilities are denoted by A_{l1}, A_{l2}, A_{l3},..., A_{lm}, then the availability of this path (represented by A_p) is equal to[1]:

$$A_p = A_{l1} \cdot A_{l2} \cdot A_{l3} \cdots A_{lm} \tag{2}$$

If we take the $-log$ of the link availabilities, finding the path with the highest availability turns into a shortest path problem.

When, for a single connection, there are $k \geq 2$ link-disjoint paths p_1, p_2,..., p_k with availabilities represented by A_{p_1}, A_{p_2},..., A_{p_k}, the connection availability can be calculated for two cases, namely: (1) fully link disjoint: these k paths have no links in common, (2) partially link disjoint: at least two of these k paths traverse at least one same link. In case (1), the availability (represented by A^k_{FD}) is:

$$A^k_{FD} = 1 - \prod_{i=1}^{k}(1 - A_{p_i}) = \sum_{i=1}^{k} A_{p_i} - \sum_{0<i<j\leq k} A_{p_i} \cdot A_{p_j} + \sum_{0<i<j<u\leq k} A_{p_i} \cdot A_{p_j} \cdot A_{p_u} + \cdots + (-1)^{k-1} \prod_{i=1}^{k} A_{p_i} \tag{3}$$

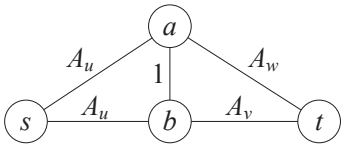


Fig. 1: Availability calculation of a pair of fully and partially link-disjoint paths.

If we use Eq. (3) to calculate the availability for the partially link disjoint case, the availability of the overlapping links will be counted more than once. To amend this, we use a new operator $\circ$, which is defined as follows:

$$X_1 \cdot X_2 \cdots X_k \circ Y = \begin{cases} \prod_{i=1}^{k} X_i & \text{if} \quad \exists X_i = Y \\ \prod_{i=1}^{k} X_i \cdot Y & \text{otherwise} \end{cases} \tag{4}$$

where $X_1, X_2, ..., X_k$ and Y represent different link availabilities. Therefore, the availability (represented by A^k_{PD}) of k partially link-disjoint paths can be calculated as follows:

$$\begin{aligned} A^k_{PD} &= 1 - \coprod_{i=1}^{k}(1 - A_{p_i}) \\ &= 1 - (1 - A_{p_1}) \circ (1 - A_{p_2}) \circ \circ \circ (1 - A_{p_k}) \\ &= \sum_{i=1}^{k} A_{p_i} - \sum_{0<i<j\leq k} A_{p_i} \circ A_{p_j} + \\ &\quad \sum_{0<i<j<u\leq k} A_{p_i} \circ A_{p_j} \circ A_{p_u} + \cdots + (-1)^{k-1} \coprod_{i=1}^{k} A_{p_i} \end{aligned} \tag{5}$$

where $\coprod$ is used to denote the $\circ$ operations of different sets. Let us use an example to describe the difference between case 1 and case 2, where k is set to 2 for simplicity. In Fig. 1 where the link availability is labeled on each link, paths $s-a-t$ and $s-b-t$ are fully link disjoint. According to Eq. (3), their availability is equal to:

$$\begin{aligned} &1 - (1 - A_u \cdot A_w) \cdot (1 - A_u \cdot A_v) \\ =&A_u \cdot A_w + A_u \cdot A_v - A_u^2 \cdot A_w \cdot A_v \end{aligned} \tag{6}$$

On the other hand, paths $s-a-t$ and $s-a-b-t$ are two partially link-disjoint paths. According to Eq. (5), the connection availability can be calculated as follows:

$$\begin{aligned} &1 - (1 - A_u \cdot A_w) \circ (1 - A_u \cdot A_v) \\ =&A_u \cdot A_w + A_u \cdot A_v - A_u \cdot A_w \cdot A_v \end{aligned} \tag{7}$$

The following theorem will formalize the intuitive notion that if a set of paths p_i with availabilities A_{p_i} have overlapping links that their total availability is less than when those paths are fully link disjoint.

Theorem 1: For given A_{p_i}, where $1 \leq i \leq k$, $A^k_{FD} \geq A^k_{PD}$.

Proof: A proof by mathematical induction:

When $k = 2$, $A^2_{FD} = A_{p_1} + A_{p_2} - A_{p_1} \cdot A_{p_2}$, and $A^2_{PD} = A_{p_1} + A_{p_2} - A_{p_1} \circ A_{p_2}$. Since $A_{p_1} \cdot A_{p_2} \geq A_{p_1} \circ A_{p_2}$ according to Eq. (4) when $0 \leq A_{p_i} \leq 1$, the theorem is correct for $k = 2$.

Assume when $k = m$ the theorem is correct:

$$A^m_{FD} = 1 - \prod_{i=1}^{m}(1 - A_{p_i}) \geq A^m_{PD} = 1 - \coprod_{i=1}^{m}(1 - A_{p_i}) \tag{8}$$

When $k = m + 1$, $A^{m+1}_{FD} = 1 - (1 - A_{p_1}) \cdot (1 - A_{p_2}) \cdots (1 - A_{p_m}) \cdot (1 - A_{p_{m+1}})$ and $A^{m+1}_{PD} = 1 - (1 - A_{p_1}) \circ (1 - A_{p_2}) \circ \circ \circ (1 - A_{p_m}) \circ (1 - A_{p_{m+1}})$. According to Eq. (8), we have:

$$\begin{aligned} &(1 - A_{p_1}) \cdot (1 - A_{p_2}) \cdots (1 - A_{p_m}) \cdot (1 - A_{p_{m+1}}) \\ \leq&(1 - A_{p_1}) \circ (1 - A_{p_2}) \circ \circ \circ (1 - A_{p_m}) \cdot (1 - A_{p_{m+1}}) \end{aligned} \tag{9}$$

Since $(1 - A_{p_m}) \cdot (1 - A_{p_{m+1}}) \leq (1 - A_{p_m}) \circ (1 - A_{p_{m+1}})$, we have:

$$\begin{aligned} &(1 - A_{p_1}) \circ (1 - A_{p_2}) \circ \circ \circ (1 - A_{p_m}) \cdot (1 - A_{p_{m+1}}) \\ \leq&(1 - A_{p_1}) \circ (1 - A_{p_2}) \circ \circ \circ (1 - A_{p_m}) \circ (1 - A_{p_{m+1}}) \end{aligned} \tag{10}$$

By combining Eq. (9) and Eq. (10), we get that $A^{m+1}_{FD} \geq A^{m+1}_{PD}$. ■

III. ABPS Complexity Analysis

In this section, we study the complexity of the ABPS problem in networks without SRLGs. For the case $k = 1$, by taking the $-log$ of the link availabilities, the ABPS problem turns into a shortest path problem, which is polynomially solvable.

Theorem 2: The ABPS problem is NP-hard for $k \geq 2$.

Proof: The case for partially link-disjoint paths can be reduced to the case of fully link-disjoint paths by a transformation such as in Fig. 2. More specifically, if we assume any link in Fig. 2, except for (s, s') and (t', t), has availability

[1]A network having node and link availabilities can be transformed to a directed network with only link availabilities, as done in [2]. Therefore, we assume the nodes have availability 1 in this paper.

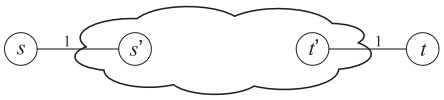


Fig. 2: Reduction of ABPS problem from partially link disjoint to fully link disjoint.

less than δ, then no link, except for (s, s') and (t', t), can be the unprotected link in the solution of the ABPS problem for the partially link disjoint case from s to t. In this context, for such δ, solving the fully link-disjoint ABPS problem from s' to t' is equivalent to solving the partially link-disjoint ABPS problem from s to t. It therefore sufficies to prove that the fully link disjoint variant for $k = 2$ is NP-hard. The proof for $k > 2$ follows analogously from the proof for $k = 2$.

We first introduce the NP-hard 3SAT problem [3] and then reduce the ABPS problem to it. The 3SAT problem is defined as: There is a boolean formula $C_1 \wedge C_2 \wedge ...C_m$, where C_i denotes the i-th clause. Each clause contains 3 variables with an OR relation. The question is whether there is a truth assignment to the variables that simultaneously satisfies all m clauses. Given a 3SAT instance, the graph construction follows

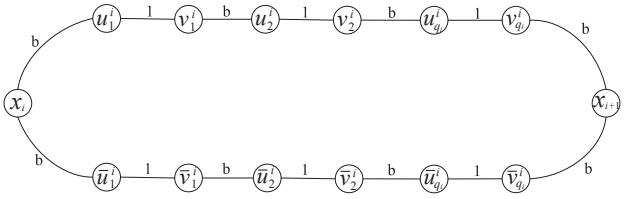


Fig. 3: A lobe for each x_i.

similarly to [4]. Assume there are n variables in the 3SAT instance. First, we create a lobe for each variable x_i, which is shown in Fig. 3, where q_i represents the number of occurrences of variable x_i in all the clauses. The availability value for each link is also shown in Fig. 3, where $0 < b < 1$. For each clause

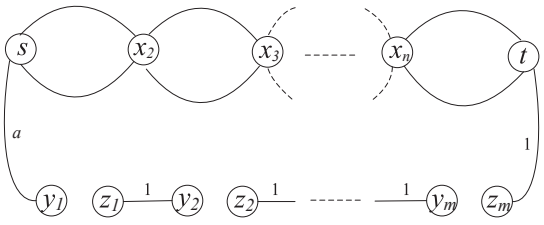


Fig. 4: Lobes for all clauses.

C_i two nodes y_i and z_i are created and a link connects z_i and y_{i+1} with availability of 1, where $0 < i < m$. We assume that $s = x_1$ and $t = x_{n+1}$. Moreover, we draw a link (s, y_1) with availability a and a link (z_m, t) with availability 1, where $0 < b < \frac{a}{2} < 1$. Fig. 4 depicts this process.

To relate the clause and variables in the constructed graph, we add the following links: (i) links (y_j, u_k^i) and (v_k^i, z_j) are added if the k-th occurrence of variable x_i exists in clause C_j; or (ii) links $(y_j, \overline{u}_k^i)$ and $(\overline{v}_k^i, z_j)$ are added if the k-th occurrence of variable x_i exists with a negation in the clause C_j. For instance, a network corresponding to 3SAT instance $(x_1 \wedge x_2 \wedge \overline{x}_4) \vee (\overline{x}_1 \wedge x_2 \wedge x_3) \vee (x_2 \wedge x_3 \wedge x_4) \vee (\overline{x}_1 \wedge \overline{x}_3 \wedge x_4)$ is shown in Fig. 5. Based on the constructed graph, which

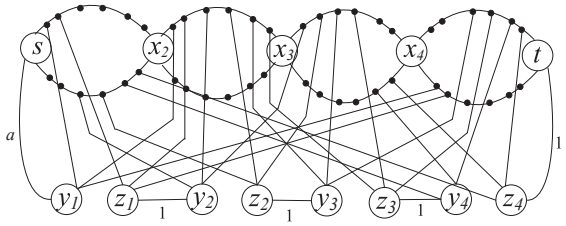


Fig. 5: Constructed graph that corresponds to $(x_1 \wedge x_2 \wedge \overline{x}_4) \vee (\overline{x}_1 \wedge x_2 \wedge x_3) \vee (x_2 \wedge x_3 \wedge x_4) \vee (\overline{x}_1 \wedge \overline{x}_3 \wedge x_4)$.

corresponds to a given 3SAT instance, we are asked to solve the ABPS problem for $k = 2$ and $\delta = a + b^q - ab^q$, where q is the sum of occurrences for each variable in all the clauses, i.e., $q = \sum_{i=1}^{n} q_i$. Because one shortest path can at most have availability a, which is less than δ, we have to find 2 link-disjoint paths. Next we will prove the fully link disjoint variant of the ABPS problem is NP-hard.

3SAT to ABPS: If there exists a truth assignment that satisfies all the clauses, then each clause j has (at least) one variable with true or (negated) false assignment to make this clause true. Therefore, an upper subpath $y_j - u_k^i - v_k^i - z_j - y_{j+1}$ or a lower subpath $y_j - \overline{u}_k^i - \overline{v}_k^i - z_j - y_{j+1}$ will be selected. By concatenating these m subpaths with $s - y_1$ and $z_m - t$ we obtain one path (denoted by p_1) with availability a. Since each variable only has one truth assignment, p_1 cannot traverse both the upper subpath and lower subpath in the same lobe. Subsequently, we can get another fully link-disjoint path p_2: For each lobe i (corresponding to variable x_i), p_2 traverses the upper (lower) subpath with availability of b^{q_i} if p_1 goes through the link of lower (upper) subpath. The availability of p_2 is $b^q = b^{\sum_{i=1}^{n} q_i}$, therefore p_1 and p_2 together have availability of $a + b^q - ab^q$, which satisfies the requirement δ.

ABPS to 3SAT: If there are two fully link-disjoint paths from s to t with availability no less than $a + b^q - ab^q$, then one path must have availability a. The reason is that if none of two paths has availability a, without loss of generality, we denote one path has availability of $a^c b^e$, where c can be either 0 or 1 indicating whether link (s, y_1) has been traversed, and $e > 0$ is the number of links which have availability b. Since there exists only one link with availability a, the other link-disjoint path has availability $a^{c'} b^f$, where c' is either 0 or 1 meaning whether link (s, y_1) has been traversed and $c' + c \leq 1$, and $f > 0$ is the number of links which have availability of b. Hence, the availability of these two paths is $a^c b^e + a^{c'} b^f - a^{c+c'} b^{e+f} < b + b < a < \delta$, when $b < \frac{a}{2}$. Based on this analysis, there must exist one path p_1 from s to t with availability a, which goes through (s, y_1) and (z_m, t) and the other links with availability of 1. To satisfy the availability requirement, there must also exist another fully link-disjoint path p_2 from s to t with availability of no less

than b^q. For each lobe, p_2 should traverse either the upper subpath or the lower subpath, otherwise p_1 and p_2 cannot be fully link disjoint. Therefore, p_2 will traverse the (entire) lower subpath if p_1 goes through link (u_k^i, v_k^i) in the upper subpath, and traverse the (entire) upper subpath if p_1 goes through link $(\overline{u}_k^i, \overline{v}_k^i)$ in the lower subpath for each lobe x_i. That is to say, p_1 cannot simultaneously traverse one link in the upper subpath and another link in the lower subpath for each same lobe. Consequently, p_1 either goes via an upper subpath $y_j - u_k^i - v_k^i - z_j - y_{j+1}$ to set variable x_i to true or via a lower subpath $y_j - \overline{u}_k^i - \overline{v}_k^i - z_j - y_{j+1}$ to set variable x_i to false for clause j, where $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$. Hence, all the m clauses can be simultaneously satisfied. ■

We proceed to study the approximability of the ABPS problem.

Theorem 3: The ABPS problem for $k \geq 2$ cannot be approximated to arbitrary degree in polynomial time, unless P=NP.

Proof: We can check in polynomial time whether a single path can accommodate the requested availability. Hence, the theorem is equivalent to: for a request $r(s, t, \delta)$ and any constant number $d > 1$, there is no polynomial-time algorithm that can find at least 2, but at most k, fully or partially link-disjoint paths from s to t with availability at least $\frac{\delta}{d}$. We prove the theorem for the fully link disjoint variant[2] of the ABPS problem for $k = 2$.

We will use a proof by contradiction and assume a polynomial-time approximation algorithm A exists for any $d > 1$. In the constructed graph based on the given 3SAT instance in Fig. 5 (also using the same notation and conditions), assume $\delta = a + b^q - ab^q$, so algorithm A can find two fully link-disjoint paths with availability at least $\frac{a+b^q-ab^q}{d}$. Next, we prove that when $0 < b < \frac{a}{2d}$, except for an exact solution, there exists no solution with availability no less than $\frac{a+b^q-ab^q}{d}$. If the exact solution is not achieved by algorithm A, according to our previous analysis, then one path must have availability of $a^c b^e$ and the other path has availability of $a^{c'} b^f$. Therefore, the availability of these two paths is equal to $a^c b^e + a^{c'} b^f - a^{c+c'} b^{e+f}$. For a given d, we have $a^c b^e + a^{c'} b^f - a^{c+c'} b^{e+f} < b + b = 2b < \frac{a}{d}$, when $0 < b < \frac{a}{2d}$ and $0 < a < 1$. Therefore, under $0 < b < \frac{a}{2d}$, except for an exact solution, any two fully link-disjoint paths cannot have availability no less than $\frac{a+b^q-ab^q}{d}$. To fulfill the assumption, algorithm A has to find two link-disjoint paths with availability $a+b^q-ab^q$. In this context, the fully link disjoint variant of the ABPS problem for $k = 2$ can be solved exactly in polynomial time, which is a contradiction. ■

IV. Shared-Risk Link Groups

In this section we will assume two types of failures/availabilities, namely Shared-Risk Link Group (SRLG) failures and single link failures/availabilities. A Shared-Risk Link Group (SRLG) [5] reflects that a certain set/group of links in a network will fail simultaneously. For instance, in optical networks, several fibers may reside in the same duct and a cut of the duct would cut all fibers in it. One duct in this context corresponds to one distinct SRLG. If each link is a single member of an SRLG, then no SRLGs exist. Hence the ABPS problem in SRLG networks includes as a special case the ABPS problem without SRLGs as discussed in the previous section. Each link can belong to one or more SRLGs, and the links which contain the same SRLG will simultaneously fail when the corresponding SRLG fails. The probability of this happening (or not) is the SRLG failure (availability) probability. We assume there are g SRLGs in the network $G(\mathcal{N}, \mathcal{L})$, and the failure probability of the i-th SRLG (represented by $srlg_i$) is denoted by π_i, for $1 \leq i \leq g$. For a particular link $l \in \mathcal{L}$, we denote by SR^l the set of all SRLGs that l belongs to. Different from [6], where all SRLG events are assumed to be mutually exclusive which means only one SRLG failure can occur at one time, we assume that multiple SRLG events may occur simultaneously. The availability of a single path should incorporate the SRLG availabilities as well as the link availabilities. Consequently, the availability of path p can be calculated as:

$$\prod_{i; srlg_i \cap p \neq \emptyset} (1 - \pi_i) \prod_{l \in p} A_l \tag{11}$$

where $\prod_{i; srlg_i \cap p \neq \emptyset} (1 - \pi_i)$ in Eq. (11) is the contribution of all the traversed SRLGs, while $\prod_{l \in p} A_l$ is the availability of path p under the condition that all its traversed SRLGs do not fail.

For example, in Fig. 6, suppose there are three SRLGs in the network with failure probability 0.1, 0.4 and 0.2, respectively, and all the links have availability 0.9. We calculate the availability of path $s - a - b - t$, which traverses 2 SRLGs ($srlg_1$ and $srlg_3$). The probability that both $srlg_1$ and $srlg_3$ do not fail is $(1 - 0.1) \times (1 - 0.2)$. Under this condition, all the links on path $s - a - b - t$ have availability 0.9 and therefore path $s - a - b - t$ has a total availability of $(1 - 0.1) \times (1 - 0.2) \times (0.9)^3 = 0.52488$.

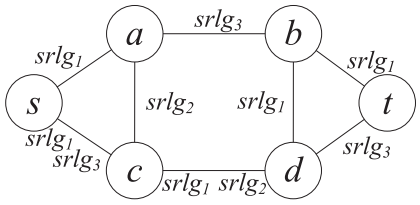


Fig. 6: Availability calculation in a SRLG network.

Next, we will prove that the single path variant of the ABPS problem in SRLG networks is NP-hard. To that end, we first introduce the Minimum Color Single-Path (MCSiP) problem, which is NP-hard [7]. Given a network $G(\mathcal{N}, \mathcal{L})$, and given the set of colors $C = \{c_1, c_2, ..., c_z\}$ where z is the total number of colors in G, and given the color $\{c_l\}$ for every link $l \in \mathcal{L}$, the Minimum Color Single-Path (MCSiP) problem is to find one path from source node s to destination node t such that it uses the least amount of colors.

[2]The partial link disjoint variant follows analogously.

Theorem 4: The ABPS problem is NP-hard in SRLG networks even for $k = 1$.

Proof: Assume a network, where all the links have availability 1 when their SRLGs do not fail, and there are g SRLGs with the same failure probability $\frac{1}{g}$. Hence, a path's availability is only determined by the number of SRLGs it traverses. If we denote one SRLG by one particular color, then the single-path ABPS problem in SRLG networks can be reduced to the MSCiP problem. ■

V. Heuristic and Exact Algorithms

Algorithm 1 $MMA(G, s, t, \delta, k, I)$

1: Find one shortest path p_1, return it if the availability requirement is satisfied, otherwise go to Step 2.
2: $ps \leftarrow 1$, $H \leftarrow p_1$, $P \leftarrow H$, $P_b \leftarrow \emptyset$ and $Q \leftarrow \emptyset$
3: **While** $ps \leq k$
4: $P \leftarrow H$
5: **For** each path $ap \in P$
6: $P_b \leftarrow P$ and $counter \leftarrow 0$
7: **While** $counter \leq I$ **do**
8: Randomly remove one link $(u, v) \in ap$ and find one shortest path $\psi_{u\rightarrow v}$ from u to v.
9: **If** it succeeds **then**
10: Replace (u, v) with $\psi_{u\rightarrow v}$ in ap, denote it as ap'
11: P_b. Remove(ap), P_b. Add(ap'), $ap \leftarrow ap'$
12: Find another link-disjoint path p_2 with P_b.
13: Return $\{p_2\} \cup P_b$ if δ is met.
14: **For** each link $(u, v) \in ap'$
15: **If** its availability is at least δ **then**
16: Q. Add((u, v))
17: **while** $(Q \neq \emptyset)$ **do**
18: $(u, v) \leftarrow$ Extract-Min(Q)
19: Find a path p_3 which shares (u, v) with ap'.
20: **If** $p_3 \notin P_b$ && $\{p_3\} \cup P_b$ satisfy δ **then**
21: Return $\{p_3\} \cup P_b$
22: **else** $H \leftarrow Max\_Availability\{H, \{p_3\} \cup P_b\}$
23: $counter \leftarrow counter + 1$.
24: $ps \leftarrow ps + 1$

A. Heuristic Algorithm

Our heuristic, called Min-Mins Algorithm (MMA) to solve the ABPS problem in both generic and SRLG networks, is presented in Algorithm 1. Since we want to use as least (and no more than k) link-disjoint paths to satisfy the requested availability, we gradually increase the number of paths.

In what follows, we explain each step of the heuristic algorithm. We assign link $l \in \mathcal{L}$ with the weight of $-log(A_l)$ ($-log(\prod_{i\in SR^l}(1 - \pi_i) \cdot A_l)$ for SRLG networks) in MMA. If a shortest path (represented by p_1) in Step 1 fails to satisfy the availability requirement, we keep it as the initial path flow. In Step 2, we use ps to record the number of already found link-disjoint paths. Initially ps is set to 1. H stores the already found ps link-disjoint paths, and it is initially assigned with p_1. While ps is no greater than k, Steps 3-24 continue finding a solution. In Step 4, we assign P with the already found paths H. Based on P from Step 5 to Step 23, we each time select one path ap from path set P. We also use a variable, denoted by $counter$ in Algorithm 1, to record the number of iterations. Initially, $counter$ is set to 0. As long as the number of iterations is less than a user given value I, Steps 7-23 are going to find a solution based on ap and path set P_b. The (sub)path from u to v found by the algorithm is denoted by $\psi_{u\rightarrow v}$. In Step 8, we randomly remove one link (u, v) from ap, and we apply a shortest path algorithm from u to v to obtain a path $\psi_{u\rightarrow v}$. By concatenating subpath $\psi_{u\rightarrow v}$ and the links of path ap except for (u, v), we obtain a new path ap'. Further, by substituting ap with ap' in P_b, we have a new path set P_b. After that, the algorithm tries to find P_b's fully link-disjoint path in Step 12. When solving the ABPS problem in SRLG networks, since each SRLG only contributes once to the path availability calculation, the link l's weight is set to $-log(\prod_{i\in\{SR^l \setminus SR^c\}}(1 - \pi_i) \cdot A_l)$ before running a shortest path algorithm in Step 12 (also the same for Step 19), where SR^c is the common traversed SRLGs between link l and path set P_b. If it fails to find p_2 or $\{p_2\} \cup P_b$ cannot satisfy the availability requirement, the algorithm tries to find a path which is partially link disjoint with ap' (in Steps 13-22). The general idea is that we first use a queue Q to store the links in ap' whose availability is no less than δ in Steps 14-16. After that, as long as Q is not empty in Steps 17-22, each time the link with the greatest availability in Q is extracted as the unprotected link (represented by (u, v)), and then we remove all the links traversed by ap' except for (u, v). Subsequently, we find one shortest path $\psi_{s\rightarrow u}$ from s to u (if it exists), and find another shortest path $\psi_{v\rightarrow t}$ from v to t (if it exists). By concatenating $\psi_{s\rightarrow u}$, (u, v) and $\psi_{v\rightarrow t}$, we can get a new path p_3, which is partially link disjoint with ap'.

The time complexity of MMA can be computed as follows. Step 1 has a time complexity of $O(N \log N + L)$. From Step 3 to Step 24, there are at most $O(I) + O(2I) + \cdots + O(kI) = O(k^2 I)$ iterations before the algorithm terminates. Steps 14-16 have a time complexity of $O(N)$ since a path contains at most $N - 1$ links and therefore Steps 17-22 consume $O(N(N \log N + L))$ time. Finally, the whole time complexity of MMA is $O(k^2 IN(N \log N + L))$.

B. Exact INLP Formulation

In this subsection, we present an exact Integer Nonlinear Program (INLP) to solve the ABPS problem. We first solve the ABPS problem without SRLGs and start by explaining the required notation and variables.

INLP notation:

$r(s, t, \delta)$: Traffic request, with source s, destination t and requested availability δ.

$A_{i,j}$: Availability of link (i, j).

g : The total number of SRLGs.

$\pi^m_{i,j}$: The occurring probability of the m-th SRLG if link (i, j) belongs to it, otherwise it is 0.

INLP variable:

$P_{i,j}^{r,u}$: Boolean variable equal to 1 if link (i,j) is traversed by path u $(1 \leq u \leq k)$ for request r ; 0 otherwise.

Flow conservation constraints:

$$\sum_{(i,j)\in\mathcal{L}} P_{i,j}^{r,u} - \sum_{(j,i)\in\mathcal{L}} P_{j,i}^{r,u} = \begin{cases} 1, & i = s \\ -1, & i = t \\ 0, & \text{otherwise} \end{cases} \quad (12)$$
$$\forall i \in \mathcal{N} \quad 1 \leq u \leq k$$

Availability constraint:

$$\sum_{u=1}^{k} \prod_{(i,j)\in\mathcal{L}} \left(1 - P_{i,j}^{r,u} + P_{i,j}^{r} A_{i,j}\right) -$$
$$\sum_{1\leq u<v\leq k} \prod_{(i,j)\in\mathcal{L}} \min\left(1 - P_{i,j}^{r,u} + P_{i,j}^{r,u} A_{i,j}, 1 - P_{i,j}^{r,v} + P_{i,j}^{r,v} A_{i,j}\right)$$
$$+ \cdots + (-1)^{k-1} \left(\prod_{(i,j)\in\mathcal{L}} \min_{1\leq u\leq k} (1 - P_{i,j}^{r,u} + P_{i,j}^{r,u} A_{i,j})\right) \geq \delta \quad (13)$$

When both the flow conservation constraint (Eq. (12)) and the availability constraint (Eq. (13)) are satisfied, an optimal solution is found by the INLP, otherwise there is no solution. Therefore, there is no objective in the proposed INLP, although one could include the objective of minimizing the number of path (or links) used. Eq. (12) accounts for the flow conservation for each of the at most k paths. For a particular u^{th} path $(1 \leq u \leq k)$, it ensures that (i) for the source node s of request r, the outgoing traffic for each request is 1; (ii) for the destination node t of request r, the incoming traffic is 1; (iii) and for an intermediate node which is neither source nor destination, its incoming traffic is equal to the outgoing traffic. Eq. (13) ensures that either the found single unprotected path or the (partially) link-disjoint paths should have availability no less than δ, according to the availability calculation of k link-disjoint paths in Eqs. (3) and (5). Since the overlapped link's availability in the partially link-disjoint calculation according to Eq. (5) can only be counted once, we take the minimum value of the variables $P_{i,j}^{r,u}$ for each link and then take the product over all the links for (partially) link-disjoint paths. We also note that Eq. (13) can simultaneously calculate the availability of the fully link disjoint variant, partially link disjoint variant and the unprotected variant. For instance when $k = 2$, Eq. (13) becomes:

$$\prod_{(i,j)\in\mathcal{L}} (1 - P_{i,j}^{r,1} + P_{i,j}^{r,1} A_{i,j}) + \prod_{(i,j)\in\mathcal{L}} (1 - P_{i,j}^{r,2} + P_{i,j}^{r,2} A_{i,j}) -$$
$$\prod_{(i,j)\in\mathcal{L}} min(1 - P_{i,j}^{r,1} + P_{i,j}^{r,1} A_{i,j}, 1 - P_{i,j}^{r,2} + P_{i,j}^{r,2} A_{i,j}) \geq \delta \quad (14)$$

When $P_{i,j}^{r,1} = P_{i,j}^{r,2}$ for all $(i,j) \in \mathcal{L}$, Eq. (14) is equal to $\prod_{(i,j)\in\mathcal{L}} (1 - P_{i,j}^{r,1} + P_{i,j}^{r,1} A_{i,j}) \geq \delta$ or $\prod_{(i,j)\in\mathcal{L}} (1 - P_{i,j}^{r,2} + P_{i,j}^{r,2} A_{i,j}) \geq \delta$, which is the availability constraint for a single unprotected path.

To solve the ABPS problem in SRLG networks, we need to slightly modify Eq. (13) (and keep flow conversation constraints Eq. (12) the same) by using $\prod_{1\leq m\leq g} \min_{u=1}^{k} \left(1 - \min_{(i,j)\in\mathcal{L}} (1 - P_{i,j}^{r,u} + P_{i,j}^{r,u} \pi_{i,j}^{m})\right)$ to multiply the left side of Eq. (13), which is the non-occurring probability of the SRLGs which at most k link-disjoint paths have traversed.

VI. Simulation Results

A. Simulation Setup

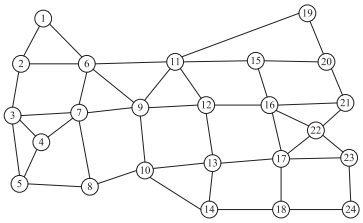


Fig. 7: USA carrier backbone network.

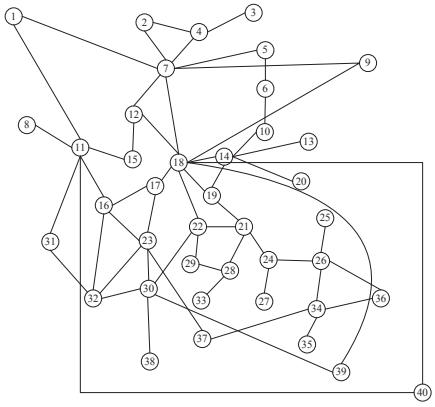
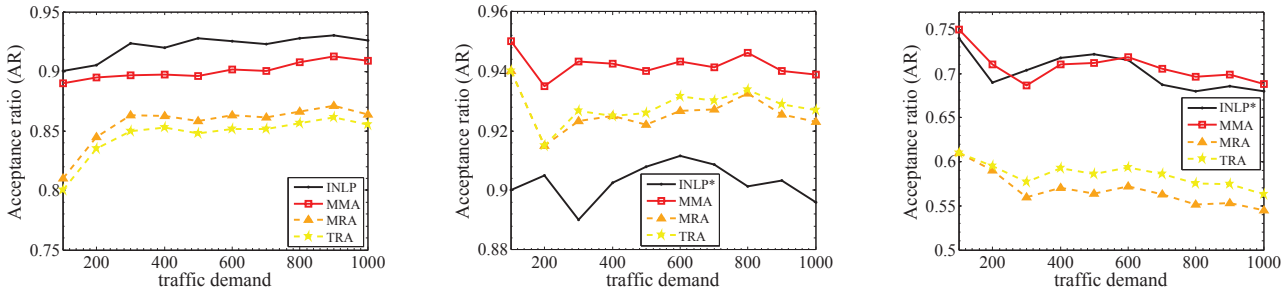


Fig. 8: GÉANT pan-European research network.

We conduct simulations on two networks, one is USANet, displayed in Fig. 7, which is a realistic carrier backbone network consisting of 24 nodes and 43 links, and the other is GÉANT, shown in Fig. 8, which is the pan-European communications infrastructure serving Europe's research and education community consisting of 40 nodes and 63 links. The simulation deals with the ABPS problem in both generic networks (i.e., without SRLGs) and SRLG networks. For generic networks, we assume the availability of fiber links is distributed among the set $\{0.99, 0.999, 0.9999\}$, with a proportion of 1:1:2. Based on the same link availabilities with generic networks, in SRLG networks we assume that there are in total 5 SRLG events with the occurring failure probabilities 0.001, 0.002, 0.003, 0.004 and 0.005, respectively. Each link has randomly been assigned to at most 3 SRLG events. For both two networks, since we want to compare the ability of finding paths for the algorithms, the capacity is set to infinity. We vary the number of traffic requests from 100 to 1000. The source and destination of each request are randomly generated, and each request has infinite holding time. The requested availability includes two cases: (i) general availability requirement case: the availability is randomly distributed among the set $\{0.98$, 0.99, 0.995, 0.997, $0.999\}$; (ii) high availability requirement



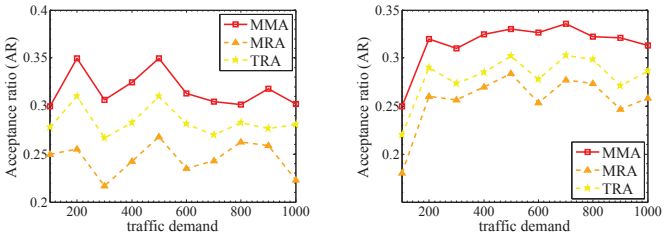
(a) GÉANT (b) USANet (c) GÉANT

Fig. 9: AR of four algorithms in generic networks: (a) general availability requirement. (USANet has been omitted since all 4 algorithms always achieved an AR of 1.); (b)-(c) high availability requirement, * max 50 mins per request.

case: the availability is randomly distributed among the set $\{0.9995, 0.9996, 0.9997, 0.9998, 0.9999\}$, by which we want to challenge the algorithm to find the feasible path under more difficult conditions. Considering the practical time complexity and the existing proposed algorithms that only focus on finding two link-disjoint paths, we choose $k = 2$. We set I in MMA to be $\lceil logN \rceil$ in these two networks (5 in USANet and 6 in GÉANT, respectively). Under the same weight allocation with our algorithm, we compare the proposed heuristic MMA and exact INLP with two counterparts: Two-step Reliability Algorithm (TRA) and Maximal-Reliability Algorithm (MRA), which are proposed in [4]. TRA first calculates a shortest path, and then calculates (if it exists) another shortest path after removing the links traversed by the first path. MRA applies Suurballe's algorithm [8] to calculate a pair of two link-disjoint paths that have minimum weight. Both of these two algorithms first apply a shortest path algorithm to check whether an unprotected path solution exits. The simulation is run on a desktop PC with 3.00 GHz CPU and 4 GB memory. We use IBM ILOG CPLEX 12.6 to implement the proposed INLP and C# to implement the heuristic algorithms.

B. Simulation Results

We first evaluate the performance of the algorithms in terms of Acceptance Ratio (AR) in generic networks. Acceptance ratio (AR) is defined as the percentage of the number of accepted requests over all the requests. We first analyze the general availability requirement case: In USANet, all the algorithms achieved an AR of 1. We therefore omit the figure of the general availability performance for USANet. However, this is not the case for the GÉANT topology. From Fig. 9(a), we can see that the performance of all algorithms is under 0.95. Since GÉANT is not as well connected as USANet is, some nodes in GÉANT only have degree one (e.g., nodes 3, 8, etc.), if an one-degree node becomes the source or the destination of a certain request, the request can only be served by partial protection (or a single unprotected path). In this context, a feasible path may not exist in GÉANT or is difficult to find, which will result in blocking. In terms of performance, the INLP achieves the highest AR. On the other hand, MMA shows a higher AR than the other two heuristics TRA and MRA (Fig. 9(a)).



(a) USANet (b) GÉANT

Fig. 10: AR of the heuristic algorithms in SRLG networks for general availability requirement.

For the high availability requirement scenario (shown in Figs. 9(b) and 9(c)), as expected, the AR of all these algorithms is lower than in the general availability requirement case. In this scenario, the INLP requires more time to find a solution, especially when a solution does not exist. In order to let the INLP return the result in a reasonable time, we set the time limit for it to serve one request to 50 minutes. Due to this reason, we can see that INLP has the lowest AR in USANet and often second highest AR in GÉANT. Meanwhile, MMA still has the highest AR in most of the cases.

The time limit for the INLP is even more constraining in the case of SRLG networks, leading to a very poor performance for SRLG networks. We have therefore omitted the results of the INLP. Since the optimal solution rarely exists in the high availability requirement case, we only provide the simulation results for the heuristic algorithms in the general availability requirement case. Moreover, to have a fair comparison, we compare our algorithms with MRA and a modified TRA [6], which is a heuristic routing algorithm proposed for the probabilistic SRLG networks. Its main idea is that after finding the first shortest path, the remaining link weights should be adjusted (We slightly change its link weight adjustment to be the same with the Step 12 of MMA for a fair comparison), and then to find another link-disjoint shortest path. Fig. 10 shows

that the proposed heuristic algorithm MMA still achieves higher AR than these two algorithms.

TABLE I: Running times per request for four algorithms (ms).

	INLP	MMA	MRA	TRA
USA Generic (General δ)	10190	0.187	0.128	0.127
GÉANT Generic (General δ)	29896	0.558	0.143	0.142
USA Generic (High δ)	79764	0.224	0.147	0.146
GÉANT Generic (High δ)	135181	0.679	0.162	0.160
USA SRLG (General δ)	$> 3.6 \cdot 10^7$	0.461	0.136	0.161
GÉANT SRLG (General δ)	$> 3.6 \cdot 10^7$	0.663	0.167	0.196

Finally, in Table I, we present the (average) running times per request for these four algorithms in both generic and SRLG networks. It shows that the INLP is significantly more time consuming than all three polynomial-time heuristics. On the other hand, MMA has only a slightly higher running time than MRA and TRA, but it pays off by having a higher AR as shown in Fig. 9. Another observation is that, for the same algorithm in the same network, the running time is higher for the high availability requirement case than in the general availability requirement case.

VII. Related Work

Zhang *et al.* [9] present a mathematical model to compute availability for different protection types (unprotected, dedicated protection and shared protection) for a set of given static traffic matrix. An Integer Linear Programming (ILP) model and a heuristic algorithm are proposed to find availability-aware paths. Tornatore *et al.* [10] address the availability design problem: to accommodate a given traffic matrix by using shared/dedicated protection paths. Song *et al.* [11] propose an availability-guaranteed routing algorithm, where different protection types are allowed. They define a new cost function for computing a backup path when the unprotected path fails to satisfy the availability requirement. She *et al.* [4] prove that in dedicated protection, finding two link-disjoint paths with maximal reliability (availability) is NP-hard. They also propose two heuristics for that problem. Luo *et al*. [12] analyze the problem of Protection With Different reliability (PWD), which is to find one unprotected path or dedicated protection path such that the cost of the whole path is minimized and the reliability requirement is satisfied. They subsequently propose an ILP to solve it exactly as well as two approximation algorithms. However, the reliability (availability) calculation in [12] is different from the aforementioned papers, and assumes a single-link failure model. Assuming each link in the network has a failure probability (=1-availability), Lee and Modiano [6] minimize the total failure probability of unprotected, partially link-disjoint and fully link-disjoint paths by establishing INLPs. They further transform the proposed INLPs to ILPs by using linear approximations.

Hu [13] proves that the SRLG Diverse Routing problem, which is to find two link-disjoint paths such that these two paths do not share any common SRLG, is NP-hard. To solve it, Hu [13] presents an exact ILP and Xu *et al.* [14] propose a trap-avoidance heuristic algorithm. However, the SRLG-Diverse Routing problem is not equivalent to the one studied for SRLG networks in this paper, due to Eq. (11). Hence, the proposed algorithms in [13] [14] cannot solve our problem effectively.

VIII. Conclusion

Connection availability can quantitatively represent the degree of how reliable a connection can carry data from a source to a destination. In this paper, we have studied the ABPS problem, which is to establish a connection over at most k (partially) link-disjoint paths, for which the total availability is no less than δ ($0 < \delta \leq 1$). We have proved that the ABPS problem in general is NP-hard and cannot be approximated in polynomial time for $k \geq 2$, unless P=NP. We have further proved that in SRLG networks, even the single-path ABPS problem is NP-hard.

We have proposed a polynomial-time heuristic algorithm and an exact INLP to solve the ABPS problem with and without SRLGs. By simulations, we have found that our heuristic algorithm outperforms two existing algorithms in terms of acceptance ratio and it only requires a slightly higher running time. On the other hand, the running time of the exact INLP is significantly larger (by several orders of magnitude) than all the heuristic algorithms.

References

[1] W. Zou, M. Janic, R. Kooij, and F. Kuipers, "On the availability of networks," in *Proc. of BroadBand Europe*, Belgium, December, 2007.

[2] G. Dantzig and D. R. Fulkerson, "On the max flow min cut theorem of networks," *Linear inequalities and related systems*, vol. 38, pp. 225–231, 2003.

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

[4] Q. She, X. Huang, and J. Jue, "How reliable can two-path protection be?" *IEEE/ACM Transactions on Networking*, vol. 18, no. 3, pp. 922–933, 2010.

[5] F. A. Kuipers, "An overview of algorithms for network survivability," *ISRN Communications and Networking*, vol. 2012, p. 19, 2012.

[6] H.-W. Lee, E. Modiano, and K. Lee, "Diverse routing in networks with probabilistic failures," *IEEE/ACM Transactions on Networking*, vol. 18, no. 6, pp. 1895–1907, 2010.

[7] S. Yuan, S. Varma, and J. Jue, "Minimum-color path problems for reliability in mesh networks," *INFOCOM 2005*, pp. 2658–2669, 2005.

[8] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.

[9] J. Zhang, K. Zhu, H. Zang, N. Matloff, and B. Mukherjee, "Availability-aware provisioning strategies for differentiated protection services in wavelength-convertible wdm mesh networks," *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, pp. 1177–1190, 2007.

[10] M. Tornatore, G. Maier, and A. Pattavina, "Availability design of optical transport networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 8, pp. 1520–1532, 2005.

[11] L. Song, J. Zhang, and B. Mukherjee, "Dynamic provisioning with availability guarantee for differentiated services in survivable mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 3, pp. 35–43, 2007.

[12] H. Luo, L. Li, and H. Yu, "Routing connections with differentiated reliability requirements in wdm mesh networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 253–266, 2009.

[13] J. Q. Hu, "Diverse routing in optical mesh networks," *IEEE Transactions on Communications*, vol. 51, no. 3, pp. 489–494, 2003.

[14] D. Xu, Y. Xiong, C. Qiao, and G. Li, "Trap avoidance and protection schemes in networks with shared risk link groups," *Journal of Lightwave Technology*, vol. 21, no. 11, pp. 2683–2693, 2003.