

# REAL-TIME PATH COMPUTATION WITH RELIABILITY CONSTRAINTS

**Jian Wang\***, **Kia Makki**, **Niki Pissinou**  
*Telecomm. & Info. Tech. Institute  
Florida International University  
Miami, FL 33174*

**Dilip Sarkar**  
*Department of Computer  
Science  
University of Miami  
Miami, FL 33124*

**Renkang Luo**, **Arnold Monitzer**, **Stefan Gianordoli**  
*Siemens ICN  
900 Broken Sound Pkwy  
Boca Raton, FL 33487*

\*Correspondence Author: Tel: +1-305-348-3709, Fax: +1-305-348-3707, Email: [jian.wang@fiu.edu](mailto:jian.wang@fiu.edu)

**Abstract:** In this study, we developed algorithms that compute reliability-guaranteed least-cost path(s) between any node pair in a network. These algorithms help carriers to reduce the risk of breaching the service reliability guarantee, and to reduce unnecessary under utilization of their network resource. Simulation results show that our heuristic algorithms achieved very good performance and reduce computational complexity.

## 1. Introduction

Connection reliability is one of the critical concerns in backbone-network planning and operation. In the traditional telecom world, five-9s, or 99.999% up-time, is often the standard requirement by customers and various government regulatory bodies. In the emerging switched optical networks [1], we expect to see more diversified customer requirements over dynamically provisioned optical circuits. Although, existing fault-management mechanisms have had help to ensure the high reliability in the past, they are not designed to directly provide reliability guarantee.

Existing fault-management mechanisms are designed to combat various connection disruptions caused by reasons such as human mistake, fiber cuts or equipment failures. Some examples of these mechanisms are the automatic protection switching (APS) in SONET rings [2] and the generalized multiprotocol label switching (GMPLS) protection [3] in optical-mesh networks. The common idea behind all these mechanisms is to reserve partial network resource, and use it to automatically reroute traffic in the presence of a failure.

Protection-switching mechanisms are usually classified into the following types [4]:

- 1+1 protection: Traffic is transmitted simultaneously on two separate paths (usually over disjoint routes) from the source to the destination. The destination simply selects one of the two paths for reception.
- 1:1 protection: There are still two paths from the source to the destination. However, traffic is transmitted over only one path at a time. When working path is cut, the source and the destination both switch over to the other protection path. The 1:1 protection allows the protection path to be shared among multiple working paths.
- M:N protection: N working paths between two nodes are protected by M backup paths. The backup paths may carry preemptable traffic. In case any non-preemptable traffic route fails, the backup path is reassigned to it.

The differences among these protection mechanisms are how much recovery resource will be allocate to protect working connections. For a given connection request, the 1+1 protection provides fastest possible protection speed and highest availability but consumes large amount of resources. On the contrary, shared M:N protection allows the most flexible and efficient resource utilization, but recovers from failures slowly. Moreover, when more than one path in the working set are affected by the same failure event, there may not be an adequate number of protection paths to accommodate all of the affected traffic.

The tradeoff between the network resource consumption and the connection reliability is more difficult to quantify than qualify [5]. Connections with exactly the same type of protection may have very different reliability values due to variations in physical distance between the source and the destination nodes, fiber/equipment situation along different paths, and so on. One the other hand, customers who lease connections from carriers care more about the service availability and cost, but less, if any, about underlying protection technologies. There is a fundamental gap between “one size fits all” fault-management mechanisms and connection reliability requirements.

In order to bridge the gap between customer requirements and the resource provisioning process, we developed algorithms that compute reliability-guaranteed least-cost path(s) between any node pair in a network, assuming each network element (fiber, transceiver, etc) has independent reliability value. These algorithms help carriers to reduce the risk of breaching the service reliability guarantee, and to reduce unnecessary under utilization of their network resource.

These algorithms are designed for general mesh networks, and it can be used within the existing GMPLS fault-management framework, while keeping all the signaling part of GMPLS unchanged [3]. These algorithms are also very fast (with complexity of only several times that of standard shortest path computation), which makes it suitable for fast provisioning in optical networks and various traffic-engineering purposes. We only consider dedicated backup resources in this study, since it is a necessary assumption to make the provisioning of each new connection independent of existing connections in the network.

The rest of the paper is organized as the following. In section 2, we will define the reliability-guaranteed least-cost routing problem. In section 4, we will briefly review related research work. In section 4, we detail our algorithm. The performance of our algorithms is evaluated in section 5 by using simulation. Finally, we conclude the paper in section 6.

## 2. Problem Specification

A network can be modeled as a graph  $G(V, E)$ , where  $V$  is the set of nodes and  $E$  is the set of vertices. The study of the network can be done by study of the underlying graph with designated parameters of the network on the node and edges of the graph. In particular, we are interested in two properties: reliability of a communication path and the associated cost of the path. Let  $P_{sd}$  be a path from node  $s$  to node  $d$ , and  $R_{sd}$  and  $C_{sd}$  be the reliability and cost of the path, respectively. Let us also denote an edge on the path as  $e_{ij}^{sd}$ , where  $i$  and  $j$  are the two end nodes of that intermediate edge, and let  $r_{ij}^{sd}$  and  $c_{ij}^{sd}$  denote the reliability and cost of the link  $e_{ij}^{sd}$ , respectively. Obviously,  $i$  equals  $s$  when  $e_{ij}^{sd}$  is the first edge of the path, and  $j$  equals  $d$  when it is the last one.

The cost of a path is the summation of individual link cost along the route. The reliability of a path is the probability that all links are available simultaneously, and is equal to the product of link reliabilities. The reliability and cost of the path  $P_{sd}$  is given by the following:

$$R_{sd} = \prod_{i=s}^{j=d} r_{ij}^{sd} \quad (1)$$

$$C_{sd} = \sum_{i=s}^{j=d} c_{ij}^{sd} \quad (2)$$

Our objective is to find a path between a node pair  $(s, d)$  such that its  $R_{sd}$  is larger than a desired value  $R'_{sd}$ , while  $C_{sd}$  is as small as possible. In the literature, this problem is referred to as the reliability-constrained least-cost path problem.

However, in some cases, there may not exist a path  $P_{sd}$  that directly satisfies the desired reliability  $R'_{sd}$ . By using additional path(s) and failure protection techniques, we may be able to meet this requirement by rerouting connections in case of failure. In case of dedicated protection (1+1 or 1:1 [4]), one backup path is used for protecting a working path, and the backup path is required to be link disjoint with the primary path. With this assumption, denoting the primary and backup paths as  $P_{sd}^1$  and  $P_{sd}^2$ , respectively, the reliability and cost of the path pair are expressed as:

$$R_{sd} = R_{sd}^1 + R_{sd}^2 - R_{sd}^1 \cdot R_{sd}^2 \quad (3)$$

$$C_{sd} = C_{sd}^1 + C_{sd}^2 \quad (4)$$

Similar to the single-path case, we study the reliability-constrained least-cost path problem for double path case when one path is insufficient to satisfy the reliability requirement.

### 3. Related Work

In the above problem specification, we broke the problem into two cases: 1) for some connection requests, one path is sufficient (unless fault-repairing time is not satisfactory); and 2) for other requests where no single path exist to satisfy the reliability requirement, we compute two paths—one serves as the primary and the other one as a backup.

The delay-constrained least-cost single-path routing problem is well studied [6-9]. Different from our first case problem, each link in the delay-constrained least-cost path problem has a delay (rather than reliability) parameter. The delay of a path is the summation of link delays, and it is bounded by a given maximum value. This delay-constrained least-cost routing problem has been proven to be NP-complete [10]. In one of the earliest studies, the author of [6] proposed two  $\epsilon$ -optimal approximation algorithms, which can produce solution with their costs less than  $1+\epsilon$  times that of the optimal solution. In 1994, Widyono [7] discovered a constrained Bellman-Ford algorithm to search for the optimal solution. Not surprisingly, all these algorithms have high time complexity when the network size is large.

To reduce the computational time, several heuristics are proposed in recent years [8, 9]. Most of them are based on the idea of constructing a combinatorial link weight from delay and cost, and then apply a standard shortest path routing algorithm. Simulations show that these algorithms not only have very low time complexities, but also achieve very high probabilities of finding the optimal solution. Among these heuristics, the approach proposed in [9] appears to have the simplest form and lowest time complexity.

Utilizing multiple paths to provide improved performance has been explored in the past for various network problems [11, 12]. For example, the minimum-cost max-flow problem [11] is somewhat similar to the problem considered here. However, we have not seen any practical solution that actually addresses the constrained minimum-cost problem. Even the problem of searching the most-reliable path pair seems to be challenging enough. Classical methods for computing disjoint paths [13] do not work here since the end-to-end reliability (equation (4) in the above) is not a linear combination of path reliabilities.

The only study we have seen to deal with the same problem we are working on is [5]. In that study, authors formulated the problem as an Integer Non-linear Programming problem. The problem is then relaxed to two Integer Linear Programming problems, and solutions are obtained when the network size is small. The approach in [5] has very high computational complexity, so it can only be used for off-line network planning applications.

## 4. Our Algorithms

### 4.1. Reliability-Constrained Least-Cost Single-Path Problem

Our approach is to convert this reliability-constrained least-cost problem to the known delay-constrained least-cost problem, and then use standard algorithms to solve it. A such approach is proposed in [5], where the negative logarithm of link reliability is taken. The equation (1) then become the following:

$$-\log(R_{sd}) = -\sum_{i=s}^{j=d} \log(r_{ij}^{sd}) \quad (5)$$

Because the reliability values are between 0 and 1, the negative logarithm of each such value is a positive number. Let's denote the  $-\log(R_{sd})$  as  $A_{sd}$ , and  $-\log(r_{ij}^{sd})$  as  $a_{ij}^{sd}$ , and refers this renaming process as variable transformation. The product reliability constraint becomes a sum constraint after this variable transformation.

We combine the variable transformation with the heuristic proposed in [9] to develop our heuristic algorithm for the reliability-constrained least-cost path problem. The basic idea of this algorithm is to combine  $a_{ij}$  and  $c_{ij}$  of each link to form a link weight  $w_{ij}$ , and then compute the shortest path according to  $w_{ij}$  with the hope that the resulted path satisfies the reliability requirement and costs as little as possible.

The  $w_{ij}$  in our heuristic is simply given as  $w_{ij} = a_{ij} + \alpha \cdot c_{ij}$ , and the magic to make it work is the  $\alpha$  value. Intuitively, when  $\alpha$  equals to zero, the resulted path from the shortest-path computation is the most-reliable path from  $s$  to  $d$ . When  $\alpha$  equals to infinity, the resulted path is the least-cost path. If the reliability constraint is between the reliability of the least-cost path and the most-reliable path, then  $\alpha$  should be between zero and infinity.

The most important contribution of [9] is to prove a theorem that can be interpreted as the following in the context of our problem: both the reliability and the cost of the resulted path monotonically decrease with the increase of the  $\alpha$ . This discovery greatly facilitates the heuristic development, since we can ensure the path reliability and reduce path cost simultaneously by tuning just one parameter  $\alpha$ . If the reliability is higher than requirement, we increase the  $\alpha$  to further reduce the cost; if the reliability is lower than requirement, we reduce the  $\alpha$  to increase the reliability. To quickly locate the appropriate  $\alpha$  value, we adopt a customized binary search algorithm also proposed in [9].

Note that this algorithm is just a heuristic, since it computes the shortest path only base on the combined link weight  $w_{ij}$ . In the section 5, we will compare the performance of this algorithm with that of the optimal solution by using experiments.

#### 4.2. Reliability-Constrained Least-Cost Link-Disjoint Double-Path Problem

So far, we know how to compute three types of paths in a network: the most-reliable path, the least-cost path, and the reliable-constrained least-cost path. When two paths are needed to satisfy the reliability requirement, we try to construct the path pair from these known types of paths. Figure 1 shows all nine-possible combinations. In order to simplify latter discussion, we will use the following abbreviation:

- MR: most-reliable path
- RCLC: reliability-constrained least-cost path
- LC: least-cost path.

	MR	RCLC	LC
MR	✘	✓	✘
RCLC	✘	✘	✘
LC	✘	✓	✘

Figure 1. Possible combinations of two paths.

Let’s start from the left top most cell, which is the “MR + MR” solution. We first find a MR in the network, and then mark all the links on this path and compute another MR using the un-marked links. Actually, once the first MR is computed, we can compute the reliability of the path by using the formula (1). Plugging the value into the formula (3), we can compute the minimum reliability requirement for the second path. Using the RCLC, we can find a cheaper (when compared with the most-reliable) second path and still meet the reliability requirement. Notice that our single-path heuristic algorithm will always check the most-reliable second path, so the “MR + MR” is naturally found by the “MR + RCLC” if it is the only solution to meet the reliability requirement.

Similarly, the “MR + LC”, the “LC + MR” and the “LC + LC” solutions are also covered by one of the constrained double-path algorithms: “MR + RCLC” and “LC + RCLC”. The RCLC algorithm cannot be used on the first path since the reliability constraint is not defined.

Eventually, the problem boils down to the comparison between two constrained double-path routing algorithms. These two algorithms differ only in how to choose the first path: whether to use the least-cost path or the most-reliable path. In the next section, we will use experiments to show the trade-offs between these two options.

## 5. Simulation study

### 5.1. Simulation Model

We designed two experiments to evaluate the performances of our proposed heuristics. The first experiment compares the performance of the single-path heuristic with an optimal algorithm, and the second experiment studies performance tradeoffs between two different double-path heuristics.

The experiments are carried out on randomly generated networking topologies using the Waxman's model [14]. We choose to use 100-node networks, since they are big enough to represent large backbone networks, and small enough to allow the simulation of brute-force optimal algorithms within reasonable time. The connectivity density of a network is measured by the average node degree, which is the average number of direct neighbors of each node. The total number of links in a network can be calculated from  $n \times g / 2$ , where the  $n$  is the number of nodes and  $g$  is the average node degree. We choose to use integer values between 4 and 10 as average node degrees to cover a range of topologies from sparsely connected to fairly densely connected topologies. Thirty networks topologies are randomly generated for each connectivity density, and they are used in our simulations. The cost of each link  $c_{ij}$  is fixed to one. The reliability of each link  $r_{ij}$  is a randomly generated variable with uniform distribution between 99% and 99.99%.

The first experiment is to measure the changes in success rate for establishing a path in a network under different path-reliability constraints. All-to-all traffic requests are assumed, and the path reliability constraints are ranged from 95% to 99.5% with step size of 0.5%. Results from the RCLC heuristic are compared with those from the least-cost path algorithm and the optimal algorithm [7].

In the second experiment, we compare the connection success rate resulted from two reliability-constrained double-path heuristics. Note that when the reliability of the first path satisfies the requirement, we do not compute the second path. Same as the last experiment, we change the path reliability from 95% to 99.5% with a step size 0.5%. For performance comparison purpose, we also used the LC and MR algorithms.

### 5.2. Simulation Results

First, we present results for single path routes. To avoid too many data and ease of visualization of the results, we show plots for average node degrees 4, 7, and 10 only. Recall that for each network, three algorithms are used to find all-to-all paths. To be more specific, for a given value of desired path-reliability, we find the percentage of all-to-all paths, discovered by a specific algorithm, that satisfy the desired reliability.

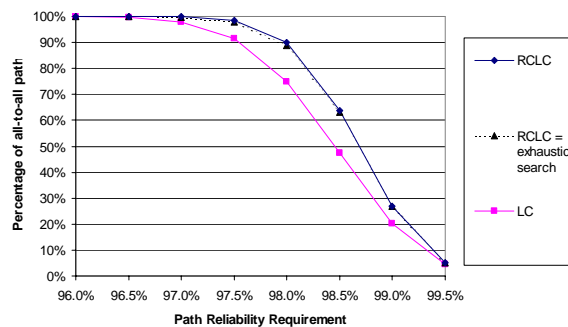


Figure 2. Randomly generated 100-node network with average node degree 4

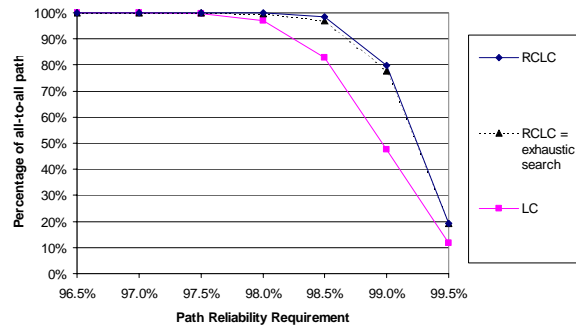


Figure 3. Randomly generated 100-node network with average node degree 7

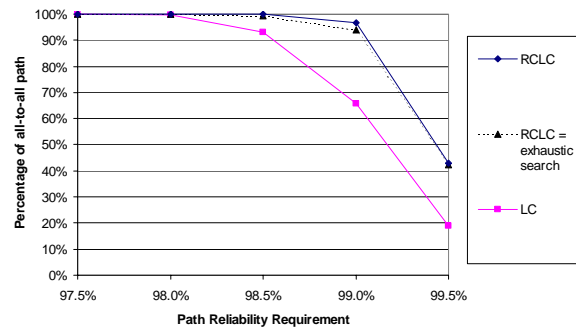


Figure 4. Randomly generated 100-node network with average node degree 10

Figures 2, 3, and 4 show simulation results about how the connection-setup-success rate changes with respect to the increase of path reliability requirement, when single-path routes are used. The lowest curve is obtained by using the LC algorithm. The highest curve is obtained by using either the RCLC or exhaustive search. The reason that the connection success rate for RCLC is always the same as exhaustive search is the following: the MR is always within the consideration of RCLC, and even exhaustive search cannot achieve reliability higher than the result of MR. The dashed curve is the percentage of solutions that found by RCLC actually has the same cost as those found by exhaustive search algorithms. It is from this curve that we can get a sense how good this heuristic algorithm is.

As expected, The RCLC algorithm has better chance to find a path that satisfies stringent reliability requirement than the LS. These general trends were consistent for networks with different connectivity densities (see Figures 2, 3 and 4). Although 100-node networks are only modest size, when average degree of the networks was 4, the probability for finding feasible solutions start to drop drastically for all algorithms when the desired path reliability is 97.5% or higher (see Figure 2). For average degree 7 networks, higher reliability constraint was easier to be met – the RCLC can still achieve almost 100% success rate when trying to setup paths under the path reliability requirement of 98%. For networks of average node degree 10, RCLC algorithm could not find every path when desired path reliability is higher than 98.5%. The results show that when the path reliability requirement is equal or higher than 99%, the effectiveness of RCLC over LC is actually reduced. This is because the path reliability requirement is so high that even the MRs cannot satisfy it.

To achieve high success rate in all-to-all connection setup when path reliability is high, multiple paths are usually required. We next present our simulation results for double paths.

Recall that for selection of two paths, we used two algorithms. One of the algorithms uses LC as the first path between a node pair. The other algorithm uses MR as the first path between a node pair. The second path between each node pair, in both the cases, is chosen in such a way that the combined reliability is just enough to satisfy the desired reliability. Before we present our simulation results for double path reliability, it is worth mentioning that for all networks, LC algorithm found fewer paths with desired reliability than the MR algorithm (see Figures 5, 7,

and 9). The reliability gain was at the cost of number of hops; average hop distance of paths obtained by the MR algorithm was larger than that obtained by the LC algorithm (plots are not shown).

It was observed that “MR + RCLC” obviously outperforms “LC + RCLC”. The observation is valid for networks of average degrees 4, 7 and 10 (see Figures 5, 7, and 9). The gain in percentage of paths with desired reliability was not free. Figures 6, 8, and 10 show that the average hop distance of paths obtained from the “MR + RCLC” algorithm is consistently higher than that obtained from the “LC + RCLC” algorithm.

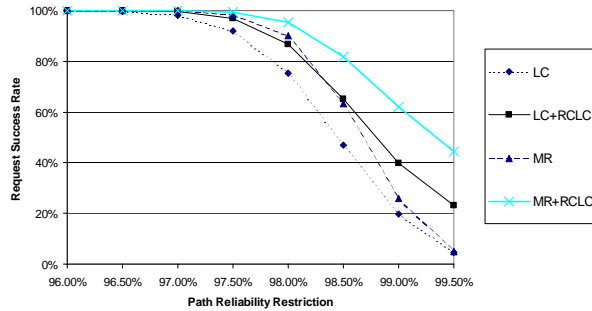


Figure 5. Double Paths on Randomly Generated 100-Node Networks (average node degree 4)

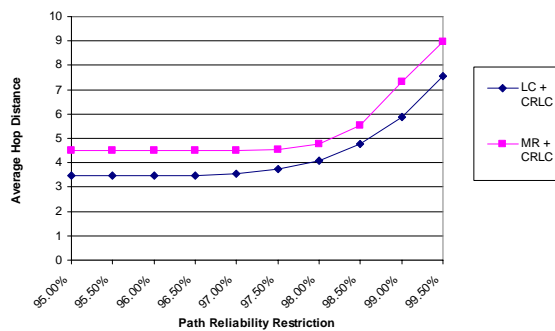


Figure 6. Double Paths on Randomly Generated 100-Node Networks (average node degree 4)

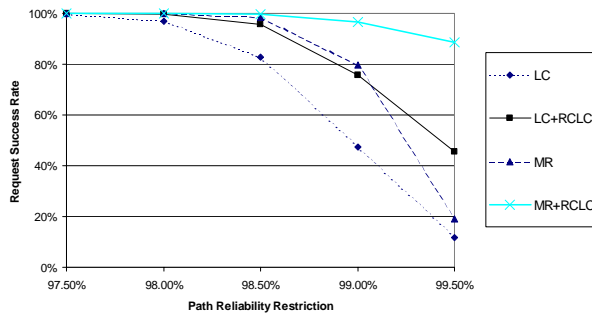


Figure 7. Double Paths on Randomly Generated 100-Node Networks (average node degree 7)

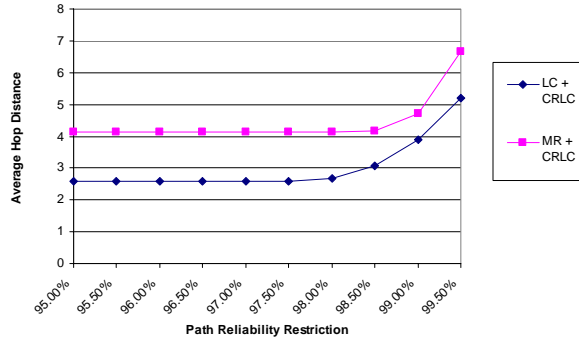


Figure 8. Double Paths on Randomly Generated 100-Node Networks (average node degree 7)

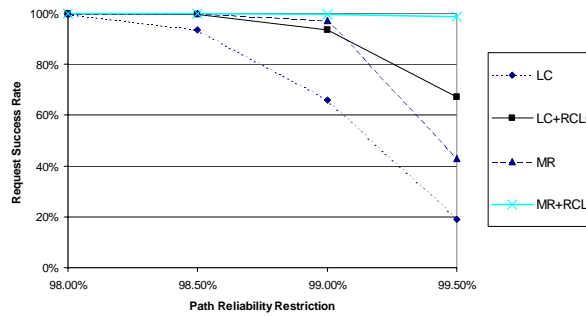


Figure 9. Double Paths on Randomly Generated 100-Node Networks (average node degree 10)

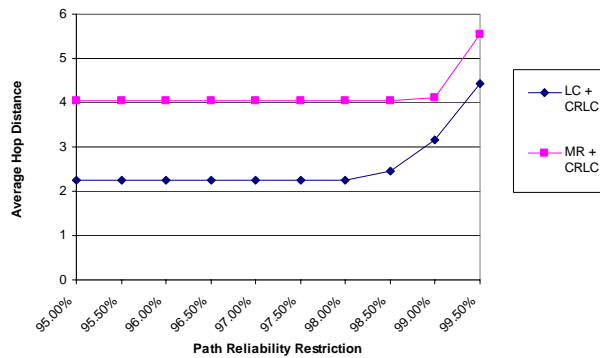


Figure 10. Double Paths on Randomly Generated 100-Node Networks (average node degree 10)

Let's look at the Figures more closely. For networks with average degree 4, single path resulted from RCLC satisfied fewer than 3% of all-to-all paths with desired reliability of 99.5%. For the same networks, "MR + RCLC" algorithms found solutions for over 20% of all-to-all paths (see Figures 2 and 5). The differences increased as average degree of the network increases. For instance, single path solution in a networks of average degree 10 could satisfy no more than 42% of the all-to-all request when path reliability is 99.5%, while the double path solution could increased this number to about 98%.

## 6. Conclusion

In this paper, we proposed a heuristic algorithm to compute the reliability-constrained least-cost path, which achieved a result close to brute force exhaustive search but works much faster. Our simulation results also show that

multi-path connection is needed when the path reliability is high since the improvement made by using more reliable single path is limited.

We, then, proposed two simple heuristic algorithms to compute reliability-constrained least-cost path pair. Results show that both of these algorithms can effectively improve the opportunity under stringent path-reliability requirements by using the second path. Especially, the “MR + RCLC” algorithm can bring the connection setup success rate to very high level when the network connectivity is sufficiently dense.

As a preliminary work, we assumed the dedicated protection to simplify the problem. If shared protections is allowed, i.e. in the case of M:N protection, then the reliability of multiple connections are correlated if they do share common backup resource. Extending this work to allow shared protection appears to be a challenging and promising future research direction. The double-path algorithms proposed in this paper use two steps to determine the path pair. Once the first path is chosen, there is no backtracking mechanism to modify it. Algorithms that can overcome this problem may deliver even better performance.

## Reference:

---

- [1] A. Banerjee, J. Drake, J. P. Lang, B. Turner, K. Kompella, and Y. Rekhter, “Generalized multiprotocol label switching: an overview of routing and management enhancements,” *IEEE Communications Magazine*, vol 39, no 1, pp 144-150, Jan 2001.
- [2] W. J. Goralski, *SONET/SDH (third edition)*, McGraw Hill, October 2002.
- [3] J. P. Lang and B. Rajagopalan, “Generalized MPLS recovery functional specification,” Work in progress, Internet Draft, draft-ietf-ccamp-gmpls-recovery-functional-00.txt, January 2003.
- [4] R. Ramaswami, K. N. Sivarajan, *Optical networks, a practical perspective (second edition)*, Morgan Kaufmann, 2002.
- [5] J. Zhang, K. Zhu, H. Zang and B. Mukherjee, “A new provisioning framework to provide availability-guaranteed service in WDM mesh networks,” Proc ICC 2003.
- [6] R. Sassin, “Approximation schemes for the restricted shortest path problem,” *Mathematics of operation research*, vol 17, no 1, pp 36-42, Feb 1992.
- [7] R. Widyono, “The design and evaluation of routing algorithms for real-time channels,” Tech. Rep. ICSI TR-94-024, University of California at Berkeley, International Computer Science Institute, June 1994.
- [8] F. Kuipers, P. Van Mieghem, T. Korkmaz, and M. Krunz, “An overview of constraint-based path selection algorithms for QoS routing,” *IEEE Communications Magazine*, vol 40, no 12, pp 50-55, Dec 2002.
- [9] G. Feng, C. Douligeris, “An efficient approximate algorithm for finding paths with two additive constraints,” *IEICE Trans. Commun.*, vol. E85-B, no. 6, June 2002.
- [10] M. R. Garey and D. S. Johnson, “Computers and Intractability: a guide to the theory of NP-completeness,” Freeman, San Francisco, CA 1979.
- [11] R. K. Ahuja, T. L. Magnati, and J. B. Orlin, *Network Flows*, Prentice Hall, Englewood Cliffs, NJ, 1993.
- [12] S. Bahk and W. El-Zarki, “Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide area networks,” in *Proc. of ACM SIGCOM*, 1992.
- [13] J. W. Suurballe and R. E. Tarjan, “A quick method for finding shortest pairs of disjoint paths,” *Networks*, vol 14, pp 325-336, 1984.
- [14] B. M. Waxman, “Routing of multipoint connections,” *IEEE Journal on Selected Areas in Communications*, vol 6, no 9, pp 1617-1622, Dec 1988.