MPLS QoS-aware Traffic Engineering for Network Resilience

Mina Amin, Kin-Hon Ho and George Pavlou

Centre for Communication Systems Research, University of Surrey, UK

Abstract: In this paper, we extend the Shortest-Distance (*SD*) routing algorithm to build resilient MPLS networks. The objective is to select paths with high connection availability, while at the same time optimising network resource consumption. The objective is achieved by using the link availability parameter in different ways in the cost function to select a path with high reliability. Having mathematically formulated the relevant traffic engineering problem and proposed algorithms to solve it, simulation results show that our algorithms provide routes with higher protection levels in comparison to the *SD* algorithm.

1. Introduction

As the Internet evolves rapidly, the volume of mission-critical and higher priority Internet applications increases and customers require not only Quality of Service (QoS) guarantees but also highly-reliable network services. The failure of network components (e.g. links, routers, etc) can lead to huge loss of data and revenue, and also can dramatically affect customers' connectivity, especially when connections are unprotected or cannot be restored. To provide network reliability against failures, many resilience (protection and restoration) methods based on Multi-Protocol Label Switching (MPLS) have been proposed [1-2]. MPLS allows network packet encapsulation at ingress Label Switch Routers (LSRs) by labelling and routing/forwarding packets along a Label Switched Path (LSP). It also provides fault detection and fault recovery, which allow effective utilization of backup paths.

A crucial issue in MPLS resilience is the creation of backup paths to protect the working paths while maintaining the required QoS even though failure happens. However, most of the existing MPLS-based resilience methods [1-2] do not take into consideration other aspects, such as the availability of network components, or parameters concerning the quality of protection, such as restoration time or packet loss when working paths are being computed. In this paper, we propose QoS-aware traffic engineering algorithms that take the above aspects into account in order to provide reliable QoS guarantees. This paper can be viewed as offline traffic engineering and bandwidth is considered as the QoS metric. Even though the concepts and methods in this paper focus on MPLS networks, they are also applicable to Generalised MPLS (GMPLS) networks.

2. Resilience methods

In general, there are two main resilience methods. The first method is to establish backup paths in order to protect working paths that may not be reliable. This method can be divided into two approaches: protection and restoration. The second method, which is the one we consider in this paper, is to establish reliable working paths. This method has so far received only little attention.

A) Resilience provisioning by establishing backup paths

1) Protection: is a static scheme in which backup LSPs are pre-computed, pre-established (before failure occurs) and pre-reserved (pre-provisioned).

2) Restoration: is a dynamic scheme in which backup LSPs have to be discovered dynamically for failed connections, so they are created and routed after failure occurs.

B) Resilience provisioning by establishing reliable working paths

In this method, resilience is provided by selecting a working path with high *connection availability. Connection availability* is the probability that the connection will be found in operating state at a random time in the future and it is determined by the availabilities of network components along the route [3]. By using the availability of network components as a parameter in the cost function of routing algorithms, a reliable working path can be found. The advantage of this approach in comparison to the backup path approach is that it provides reliable working path with appropriate and sufficient protection level for traffic requests, which minimizes the use of backup paths that consume extra network resources.

3. Availability analysis

The availability of a component is the fraction of time the component is "up" during the entire service time. In this paper, we assume that different network components fail independently. If a traffic request is carried by a single path, its availability is equal to the path availability. The path availability, denoted by A_{path}, can be calculated based on the known availabilities of the network components along its route [3]. The path is available only when all the network components along its route are available. In this paper, we consider links as the only network components but it is straightforward to incorporate other network components, such as routers. Thus, A_{path} is equal to the product of all the link availabilities along the path. Suppose the path is composed of links $l_1, l_2, ..., l_n$, then the end-to-end path availability is calculated as follows:

$A_{path} = A_1 \times A_2 \times ... \times A_n$, where A_l is the availability of link l , $1 \le l \le n$. (1)

4. Failure Impact

Two important Failure Impacts (FI) are packet loss and recovery time. Experiments in [4] show that if the failed link is close to the node responsible for re-routing, FI are less severe than in the cases where the failed link is far from the re-router node. Therefore the number of hops between the node detecting the failure and the node responsible for re-routing is an important factor of FI.

Backup LSP types depend on which node along the LSP takes the rerouting decision and this is called recovery scope [4]. In the global scope, the ingress node is responsible for the route recovery by selecting an alternate disjoint backup path, while for the local scope, the LSR at the head of the failed link reroutes the traffic from the failed link to the backup path. In the global scope, only one backup path per working path needs to be created and maintained, while in the local scope, multiple backup paths creation and maintenance are required. On the other hand, local scope leads to a minimum FI since the number of hops in this case is zero. In the global scope, the FI depends on the number of hops. The main drawback is that the number of hops is not known in advance because obviously it is not known which link will fail. However, the link availability can be used to estimate these distances in a probabilistic manner. Therefore, in the case of global recovery, the routing algorithm can assign more weight to the low availability links that are far from the ingress node, so as to be able to avoid them.

5. Extending Shortest-Distance (*SD*) routing algorithm using link availability parameter

We extend the *SD* algorithm to select a reliable path with high availability for each traffic request. The *SD* routing algorithm [5] is basically a shortest-path algorithm with the distance or cost function defined as

$C(p) = \sum_{l=1}^{n} c_l = \sum_{l=1}^{n} \frac{1}{R_l}$, where R_l is the residual bandwidth of link l . (2)

The above cost function balances the objectives of minimizing resource consumption and improving load balancing in an appropriate way. However, it does not consider the objective of improving network reliability. In the following sections, various extensions of the *SD* cost function are proposed to provide reliability. They use the link availability parameter in two ways. One way is to use link availability as a policy to define the cost function. The other way is to use it directly in the cost function. In both cases, Constrained Shortest Path First (CSPF) algorithm is used to find the bandwidth constrained shortest path according to the cost function. We define low and high availability link as the ones with $A_l < 0.9999$ and $A_l \ge 0.9999$ respectively.

A. Using availability as policy

A1. *Availability Policy_Load Balancing (AP_LB)*: In cost function (3), in comparison to the *SD* (2), a constant , α , is associated with the residual bandwidth of high availability links, while for the low availability links the cost function is the same as the *SD.* In this paper, we set α to 2, which has sufficient impact on the cost function. By using this policy, the links with high availability are more likely selected as they provide better availability than the low availability ones.

$$C(p) = \sum_{l=1}^{n} \begin{cases} \frac{1}{R_l^{\alpha}}; & if \quad A_l \ge 0.9999 \\ \frac{1}{R_l}; & \text{otherwise} \end{cases} \qquad (3)$$

A2. *Availability Policy_Hop count (AP_H)*: In cost function (4), a hop count penalty, 2^{l-1}, is associated with the low availability links while for the high availability links the cost function is the same as the *SD* (2). Moreover, this cost function attempts to avoid the low availability links at far distances, e.g. if a low availability link is 4 hops away from the source node, its cost is $\frac{2^{4-1}}{2^{2-1}} = 4$ times more than the link which is 2 hops away from the source node but both have the same availability. The reason for that, as mentioned in section 4, is to minimise the FI if global recovery is used for restoration.

$$C(p)=\sum_{l=1}^{n}\begin{cases}\frac{1}{R_l}; & if \quad A_l \geq 0.9999\\ \frac{2^{l-1}}{R_l}; & \text{otherwise}\end{cases} \qquad (4)$$

B. Using availability in cost function

In the following algorithms, *k*-Shortest-Distance *(k-SD)* paths are pre-computed by modifying the algorithm proposed in [6]. Among the *k* paths, two strategies are applied.

B1. *k-SD_Availability (k-SD_A)*: If we compute the logarithm of both sides of equation (1) and multiply it by –1, since A_l is a value between 0 and 1, *-logA$_l$* will be a positive value. We compute the availability of each path using cost function (5) and select the one with the maximum availability.

$$C(p)=\sum_{l=1}^{n}-\log A_l \qquad (5)$$

B2. *k-SD_A Hop count (k-SD_AH)*: In cost function (6), a hop count penalty, 2^{l-1}, is associated with each link in addition to its link availability. This algorithm not only attempts to find a path with high availability, but also, like *AP_H*, it minimises the FI.

$$C(p)=\sum_{l=1}^{n}2^{l-1}\times(-\log A_l) \qquad (6)$$

6. Network model and Simulation results

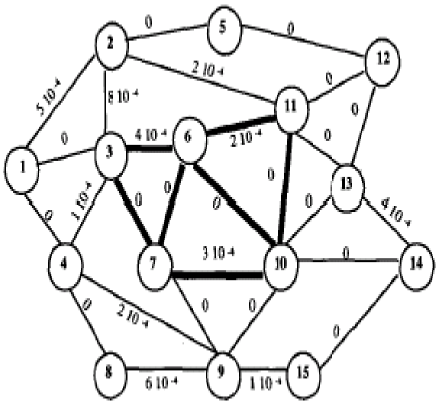

Figure 1. Network Topology

In a network, each link *l* is associated with a vector (*R$_l$*,*A$_l$*). Each traffic request *i* is defined by (s_i, t_i, B_i), where *s$_i$* is an ingress router, *t$_i$* is an egress router and *B$_i$* is the requested bandwidth. The path selected for each traffic request will meet the requested bandwidth of the traffic. The proposed algorithms in section 5 were simulated for the network topology shown in figure 1, which has been used in [1,4]. The topology has 15 nodes, 28 links and 7 nodes are identified as traffic source and destination nodes (nodes 1, 2, 4, 5, 9, 13 and 15). The capacity of links is 1200 and 4800 (bolded lines) units and each link is bi-directional. The number on each link represents link failure probability (*F$_l$*). We convert link failure probability to link availability (*A$_l$*) by employing the formula *A$_l$ =1-F$_l$*. We assume that a link needs to be protected if $A_l < 0.9999$. There are 9 links to be protected, which correspond to 32.1% of the network links. For the rest of the links, *A$_l$*=1. Each simulation graph shows the algorithm performance for 20 trials, each of which has different uniformly distributed traffic demand, randomly generated between 50 and 200, and different number of source and destination nodes. The performance metrics to evaluate our algorithms are:

Number of Links to be Protected (NLP): Figure 2 shows an analysis of the average number of links to be protected per LSP. The figure shows that by applying *AP_LB*, the NLP per LSP sharply decreases (78% less on average)

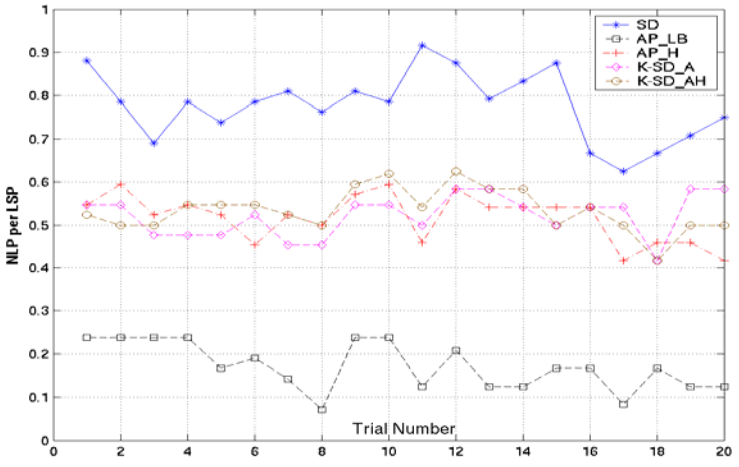

Figure 2. Number of Links to be Protected

and for the other algorithms (*AP_H*, *k-SD_A*, *k-SD_AH*), the average decrease is about 33% in comparison to the *SD* algorithm. Therefore, our algorithms reduces the number of links that need to be protected significantly, which means that resource consumption for establishing local backup paths (if necessary) can be significantly decreased.

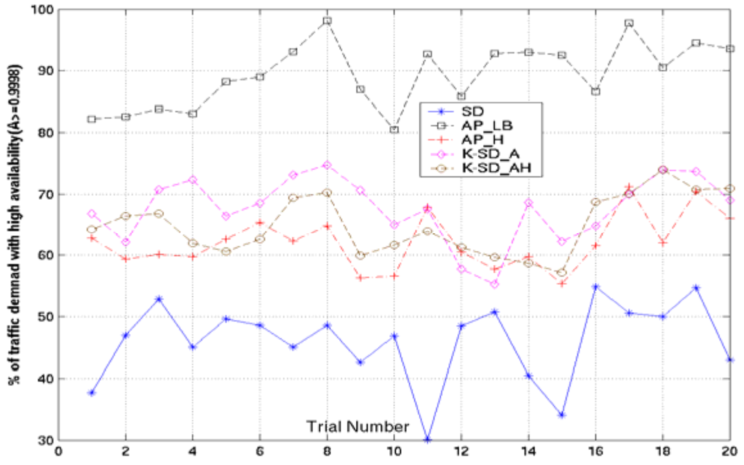

Figure 3. Network Protection Degree

Network protection degree (NPD): Figure 3 shows an analysis of *NPD*, which is the percentage of the traffic demand routed through paths with high availability. We assume that a path has high availability if $A_{path} \geq 0.9998$. The figure shows that by applying *AP_LB*, the *NPD* sharply increases (93% more on average) and for the other algorithms (*AP_H*, *k-SD_A*, *k-SD_AH*), the average increase is about 41% in comparison to the *SD* algorithm. Therefore, our algorithms increase the network protection degree and provide reliable paths for traffic requests.

Summarizing the performance of all the proposed algorithms, *AP_LB* achieves the best performance, which is much better than the performance of the *SD* algorithm in terms of providing more reliable working paths, which reduces the use of extra network resources for backup paths. However, there is a trade off between the reliability of the working path and its resource consumption. In fact, *AP_LB* may choose a longer working path in comparison to the working path computed by the *SD* algorithm. However, *AP_LB*'s working path is reliable, while *SD*'s working path needs a backup path for reliability. Therefore, the resources consumed for *AP_LB*'s working paths are comparable with those for the working and backup paths of the *SD* algorithm. Hence, *AP_LB* can offer reliable working paths without penalising the resource consumption. In addition, the other three algorithms (*AP_H*, *k-SD_A*, *k-SD_AH*) can improve reliability at the expense of slightly higher resource consumption in comparison to the resources consumed only for working paths computed by the *SD* algorithm.

7. Conclusions

In this paper, we extend the *SD* algorithm by using link availability parameter to achieve resilience of the working path. Simulation results show that by using our proposed algorithms, traffic requests are routed through high availability paths and less number of links need to be protected in comparison to the *SD* algorithm. Therefore, backup resource consumption can be significantly reduced by using the proposed algorithms. Also, the proposed algorithms allow ISPs to apply better QoS routing strategies to increase connection reliability for their customers.

References.

[1] M. Kodialam, T.V. Laksham, "Dynamic Routing of Restorable Bandwidth-Guaranteed Tunnels Using Aggregated Network Resource Usage Information", *IEEE/ACM Transactions on Networking*, June 2003

[2] C. Huang, V. Sharma, K. Owens, S. Makam, "Building reliable MPLS Networks using a path protection mechanism", *IEEE Communication Magazine*, March 2002

[3] J. Zhang, K. Zhu, H. Zang, B. Mukherjee, "A New Provisioning Framework to Provide Availability-Guaranteed Service in WDM Mesh Networks", *Proceedings of the IEEE International Conference on Communications (ICC)*, May 2003

[4] E. Calle, J.L. Marzo, A. Urra, "Evaluating the probability and the impact of a failure in GMPLS based networks", *Design of Reliable Communication Networks (DRCN)*, October 2003

[5] Q. Ma, P. Steenkiste, "On Path Selection for Traffic with Bandwidth Guarantees", *IEEE International Conference on Network Protocols (ICNP)* 1997

[6] D. Eppstein, "Finding the *k* Shortest Paths", *35th IEEE Symposium Foundations of Computer Science.*, 1994