Improving Survivability through Traffic Engineering in MPLS Networks

Mina Amin, Kin-Hon Ho, George Pavlou, and Michael Howarth
Centre for Communication Systems Research, University of Surrey, UK
Email:{M.Amin, K.Ho, G.Pavlou, M.Howarth}@eim.surrey.ac.uk

Abstract

The volume of higher priority Internet applications is increasing as the Internet continues to evolve. Customers require Quality of Service (QoS) guarantees with not only guaranteed bandwidth and delay but also with high availability. Our objective is for each estimated traffic flow to find a primary path with improved availability and minimum failure impact while satisfying bandwidth constraints and also minimizing network resource consumption. We devise a heuristic algorithm with four different cost functions to achieve our objective. Our approach can enhance availability of primary paths, reduce the effect of failure and also reduce the total resource consumption for both primary and backup paths.

1. Introduction

The steady growth in the use of computer networks for higher priority Internet applications encourages service providers to offer new services that depend on a committed Quality of Service (QoS); these services require continuous network availability in the presence of various failure scenarios. Hence, network survivability, which refers to the ability of a network to maintain uninterrupted service regardless of the scale, magnitude, duration and type of failures, is an important issue. We investigate survivability in Multi-Protocol Label Switching (MPLS) networks. Many MPLS survivability methods have been proposed [1-2]. A fundamental consideration in the design of an MPLS survivable network is the creation of backup paths to protect the primary paths from failure while preserving the required QoS as has been considered in proposals [1-2]. However, other existing MPLS-based survivability methods [3] take into consideration aspects such as availability of network components and failure impact parameters during the computation of the primary path.

Poor routing of primary paths might route traffic through low availability links, leading to higher probability of failure occurrence and as a result more failure consequences such as recovery time and packet loss. Therefore, we address an Offline Traffic Engineering Survivability Design (OTESD) problem that takes into account the network component availability and failure impact parameters in order to provision a survivable network. Our goal is summarised as follows: given a physical topology, aggregate estimated traffic flows and estimated link availability, for each traffic flow find a primary path with enhanced availability and minimum failure impact while preserving the flow's bandwidth requirement and optimising the use of network resources. To solve the OTESD problem, we adopt a dual approach in which we first find primary paths with improved availability and low failure impact. We subsequently provide backup paths for those traffic flows where a sufficiently high availability primary path cannot be found. Therefore our approach is a kind of protection scheme. Our work is motivated by the survivability method proposed in [3], however [3] is based on online routing while we consider offline Traffic Engineering (TE) for network provisioning in order to achieve considerable improvement in survivability performance and better resource utilization. Building a survivable network through offline TE has also been investigated in [7], in which an algorithm to set link weights was proposed for IP backbone networks so as to mitigate the effects of failure. However, the OTESD problem has not yet been addressed in MPLS networks and is therefore the subject of this paper. We believe no previous work has taken into account availability parameters when solving survivability design problem through offline TE.

2. Background

Survivability methods [1-2] can be divided into two basic approaches. The first approach, called protection, is a pre-determined failure recovery scheme in which at the same time as the primary path is routed between the source and the destination, the backup path is also provisioned to forward the traffic if the primary path fails. In the second approach, called restoration, first a primary path is set-up between the source and the destination; and then, after failure occurs, a backup path is discovered dynamically to restore the traffic. In this paper we use protection to achieve our objective.

Backup path types depend on which router along the primary path takes the rerouting decision, and this is called recovery scope [2]. In the *global* scope, the ingress node always takes responsibility for fault recovery when a Fault Indication Signal (FIS) arrives (any message sent to indicate that failure has occurred is called a FIS). The advantages of this scope are firstly that the backup path can be selected from links anywhere in the entire network, so the network spare resources are used efficiently, and secondly that only one backup path needs to be set up per primary path. However, since a FIS has to be propagated all the way back to the ingress node, this method has high recovery time and packet loss. In the *local* scope, the Label Switch Router (LSR) at the head of the failed link switches the traffic from the broken link to the backup path. Since a FIS is not needed, this scope has a faster recovery time and reduced packet loss in comparison to global scope. On the other hand, creation and maintenance of multiple backup segments is required, resulting in inefficient utilization of resources and increased complexity [3].

The effects of network failure can be evaluated in terms of recovery time and packet loss. The recovery time (T_{REC}) is defined as the period of time between fault detection and the traffic restoration to the corresponding backup path. Recovery time consists of a number of phases as follows [3]. a) Detection Time: the time required for fault detection. b) Hold off time: the waiting time before triggering the fault recovery process in case lower layers can overcome the fault faster. c) Notification Time: the time required to convey fault information to the node responsible for rerouting the traffic (for example, by transmitting a FIS). d) Switchover Time: the time required to redirect the traffic from the primary path to the backup path.

All these phases are followed by a global scope protection scheme; a local scope scheme does not have a notification phase.

Packet loss (P_{LS}) is defined as the total number of packets lost during T_{REC} . Since packet loss depends on the recovery time, a longer recovery time leads to more packet loss. Hence, in order to reduce the failure impact, we only need to reduce the recovery time.

Survivability provisioning by establishing backup paths has a number of limitations. Firstly, it may overuse network resources for establishing these paths if the network is not well provisioned. Secondly, it requires signalling overhead for backup path establishment, failure detection and notification. Finally, it requires traffic switching from primary path to backup path and then from backup path to primary path, which may cause network oscillation. These survivability methods thus lead to high overall overhead. Our dual approach, described below, attempts to minimize the aforementioned limitations.

3. Network availability analysis

The availability of a component i (A_i) is the fraction of time the component is operational ("up") during the entire service time [4]. A network component's availability is a relatively static value. Typical data on network components (transmitter, receiver, fiber link) failure rate and repair times can be found in [9]. In addition, the availability of a component can be calculated by reliability prediction models such as Telcordia [8].

If the traffic flow t is carried by a single path, its availability, denoted by α_t , is equal to the availability of the path the flow traverses. If we denote path availability by A_{path}, we have:

$$\alpha_t = A_{path} \qquad (1)$$

A_{path} can be calculated based on the known availabilities of the network components along its route [4]. Suppose the path is composed of n links, then the end-to-end path availability is calculated as follows:

$$A_{path} = A_1 \times A_2 \times ... \times A_n = \prod_{l=1}^{n} A_l \qquad (2)$$

where A_l is the availability of link l.

4. The OTESD problem

Definition 1: We distinguish between low and high availability links. In this paper, we define low availability links as $A_l < 0.9999$ and high availability links as $A_l \geq 0.9999$. Definition 2: we define the Link Protection Requirement (*LPR*) as a binary value for each link to indicate whether the link should be protected or not. For low availability links we set *LPR=1* and for high availability links *LPR=0*. Definition 3: We distinguish between low and high availability paths. In this paper, we define low availability path as $A_{path} < 0.9998$ and high availability path as $A_{path} \geq 0.9998$ respectively.

4.1. Dual approach

Our approach consists of two phases: (1) a preventive phase, and (2) an impact minimisation phase.

In the preventive phase, TE is used to map the estimated traffic flows onto the existing physical network topology in the most effective way to improve network survivability while optimising resource utilization. In order to improve the availability of the primary path our TE routes the traffic mostly through the high availability links by taking into account the link availability parameter during the computation of primary routes. This objective is achieved by using the link availability parameter in different ways in the cost function of the Shortest Distance routing algorithm [5] as explained in section 5. However, it is not always possible to avoid using the low availability links. In fact, in some cases traffic may be routed through primary paths with low availability links. In this case, we perform the impact minimization phase. Recovery time is minimized by reducing the time needed for each phase of the recovery as follows. Detection and switchover time depend on the technology used and cannot be easily modified. Hold off time depends on the lower layers recovery scheme and can be set up between (0-50ms). Therefore, the notification time seems to be the key factor for minimizing the recovery time. The notification time depends on the propagation time of a FIS per link and on the Notification Distance (*ND*) [3]. *ND* is defined as the number of links between the node that detects the failure and the node that reroutes the traffic. Since the propagation time depends only on the link transmission rate, notification time can only be decreased by reducing the *ND*. In local scope the distance is zero, so it is the optimal case. But as we discussed in section 2, the global scope is more efficient and more scalable. However, in the global scope the distance is not known in advance because obviously it is not known which link will fail. Therefore in the impact minimization phase, we use the link availability parameter to estimate the distance in a probabilistic manner. Moreover, if we cannot find a sufficiently high availability primary path for the traffic flow, we provide a global link-disjoint backup path for the corresponding primary path.

4.2. The OTESD Problem Formulation

We formulate the OTESD problem as a multi-objective optimisation problem. A solution of OTESD computes a primary path for each traffic flow, which yields the best value for one or more objective functions. The following assumptions are given:
1. *G=(V,E,A,C)*, the physical network topology where V is the set of nodes, E is the set of links, $A: E \to (0,1)$ is the availability of each link (where (0,1) denotes the set of real numbers between 0 and 1), $C: E \to Z^+$ specifies the total physical capacity on each link. 2. $T = (t = \langle s, d, B_t \rangle)$, the traffic matrix, which is a set of estimated traffic flows, where s is the source , d is the destination , and B_t is the Bandwidth requirement of the traffic flow t. We assume that aggregate estimated traffic flows have a constant bit rate pattern due to statistical multiplexing. The traffic matrix can be measured or estimated.

We consider the following objectives: (1) Improve the primary paths' availability, (2) Minimise failure impact and (3) Minimise the total Resource Consumption. These objectives are optimised subject to the solution satisfying the link capacity constraints.

4.3. A heuristic algorithm

To implement our approach, we use the following heuristic algorithm.

Primary path provisioning procedure:

Step 1: sort the traffic flows in descending order based on their bandwidth requirements and choose one at a time in that order. **Step 2**: remove the links with the residual bandwidth less than the traffic flow bandwidth requirement to ensure that the remaining links can guarantee the bandwidth requirement. **Step 3**: compute the primary path, using the cost functions proposed in section 5. **Step 4**: once the primary path is found, allocate the requested bandwidth on the path. **Step 5**: consider the next traffic flow and repeat step 2 to 5 until all the traffic flows have been considered.

Backup path provisioning procedure:

Step 1 is the same as step 1 of the primary path provisioning procedure. **Step 2**: compute the primary path availability of the corresponding traffic flow according to equation (2), if it is a low availability primary path go to the next step, otherwise go to step 5. **Step 3**: remove the links with the residual bandwidth less than the traffic flow bandwidth requirement. **Step 4**: compute a global link disjoint backup path according to the Shortest Distance cost function (3) by using the Dijkstra routing algorithm and allocate the requested bandwidth on the path. **Step 5**: consider the next traffic flow and repeat step 2 to 5 until all traffic flows have been considered.

5. Proposed cost functions

The Shortest Distance (SD) algorithm is a shortest-path algorithm with the cost function defined as:

$$C(p) = \sum_{l=1}^{n} c_l = \sum_{l=1}^{n} \frac{1}{R_l} \qquad (3)$$

Where c_l is the link cost and R_l is the residual bandwidth of link l ($1 \le l \le n$). The shortest path can be found by the Dijkstra routing algorithm using the corresponding cost function. The shortest path is the path with minimum path cost denoted by *C(p)*. The above cost function balances the objectives of minimising resource consumption and improving load balancing. However, the *SD* cost function does not consider the objective of improving path availability. Hence, it may result in using low availability primary paths. Hence, we propose several extensions of the *SD* cost function to achieve our dual approach objectives. Our proposals use the link availability parameter in the following two ways.

A. Using link availability as a threshold:

1) Availability Threshold 1 (*AT1*): The cost function (3) is modified by including a hop count penalty, 2^l, for the low availability links. For the high availability links the cost function remains the same as SD (3). (We experimented the hop count penalty with values other than 2 and we observed no significant changes in the results). In this approach, higher costs are assigned to low availability links than the high availability ones. Moreover, the low availability links which are far away from the ingress node are penalized more in order to reduce the failure impact as explained in section 4.1. In this way the failure notification distance (*ND*) is decreased which results in reduced recovery time (T_{REC}) and, therefore, reduced packet loss (P_{LS}). Our *AT1* cost function is defined as:

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

2) Availability Threshold 2 (*AT2*): Our second cost function, *AT2*, is defined as follows.

$$C(p) = \sum_{l=1}^{n} \begin{cases} \dfrac{2^{l-1}}{R_l^{\beta}} & if \ A_l \geq 0.9999 \\ \dfrac{2^{l-1}}{R_l} & \text{otherwise} \end{cases} \qquad (5)$$

In comparison to SD (3), a hop count penalty, 2^{l-1}, is applied both to high availability and low availability links in order to assign higher cost to the links located far from the ingress node. This gives lower costs to shorter paths. A constant, β, is associated with the residual bandwidth of high availability links. We set β to 2, which experiments show that has sufficient impact on the cost function. By squaring the residual bandwidth of the high availability links less cost is assigned to them in comparison to the low availability ones.

B. Using link availability in the cost function:

1) *K-SD-AD*: First of all, K-Shortest-Distance (K-SD) paths are computed by modifying the algorithm proposed in [6] and using cost function (3). Then among the K paths, the path with the minimum cost function according to (6) is selected as the final path. We associate a hop count penalty, 2^{l-1}, with each link in addition to its link availability function. Low availability links located far away from the ingress node are therefore penalized more. As a result, this cost function combines the objectives of improving primary path availability with minimising failure impact.

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

According to the order of the cost functions used in this case, the path selected here first optimises the resource consumption and load balancing objectives and then optimises the path availability and failure impact objectives.

2) *K-AD-SD*: First of all, K-shortest (K-AD) paths are computed by modifying the algorithm proposed in [6] and using cost function (6). Then among the K paths, the path with minimum cost function according to (3) is selected as the final path. In fact, the cost function used in this approach is the same as *K-SD-AD* but in the opposite order. Hence, the path selected in this case optimizes the objectives in the opposite order.

6. Performance evaluation

We evaluate our heuristic algorithm with the four proposed cost functions through simulation using the following four performance metrics.

1. Network Protection Degree (NPD):

$$NPD = \frac{\sum_{t \in T} (B_t \mid \alpha_t \geq \varepsilon)}{\sum_{t \in T} B_t} \qquad (7)$$

where T is the total number of traffic flows, bandwidth is denoted by B_t and $\varepsilon = 0.9998$ in this paper. If a cost function results in a high *NPD* value it implies that the cost function performs better regarding improving the primary path availability which is the first objective.

2. Failure Impact Degree (FID):

$$FID = \frac{\sum_{t \in T} (B_t \mid ND_t > 1)}{\sum_{t \in T} B_t} \qquad (8)$$

If a cost function results in a low *FID* value it implies that it performs better regarding minimising the effects of failure which is the second objective.

3. Number of Links to be Protected (NLP):

$$NLP = \frac{\sum_{t=1}^{T} \sum_{l=1}^{n_p} (LPR_l \times x^t{}_l)}{T} \qquad (9)$$

where $x^t_l=1$ if traffic flow t has been assigned to link l; otherwise $x^t_l=0$. Also, n_p is the number of links on the primary path and *LPR* is the parameter defined in definition 2 (in section 4). If a cost function results in a low *NLP* value, it reduces resource consumption by only establishing local backup paths for low availability links, which is the third objective.

4. Resource Consumption (RC):

$$RC = \sum_{t=1}^{T} n_t \times B_t \qquad (10)$$

where n_t is the number of links on the traffic flow's path and B_t is the bandwidth required for each traffic flow. We compute the *RC* for two cases. In the first case, we only compute the resources consumed by the primary paths. Therefore, if we denote the number of links on a single traffic flow's primary path by n_p, then for this first case $n_t=n_p$. In the second case, we provide a pre-established, pre-allocated global link-disjoint backup path for each low availability primary path to attempt to maximize availability for 100% of the traffic flows. We assume that the required bandwidth is dedicated to each backup link and no resource sharing is considered. In this way we can evaluate the total *RC* (consisting of high and low availability primary paths and backup paths for low availability primary paths). Therefore, if we denote by n_b the number of links on the traffic flow's backup path provided for the low availability primary paths, then for the second case $n_t=n_p+n_b$. If a cost function provides low total *RC* value it performs better regarding minimising the total *RC* by establishing global backup paths (if necessary) for protecting the low availability primary paths, which is the third objective.

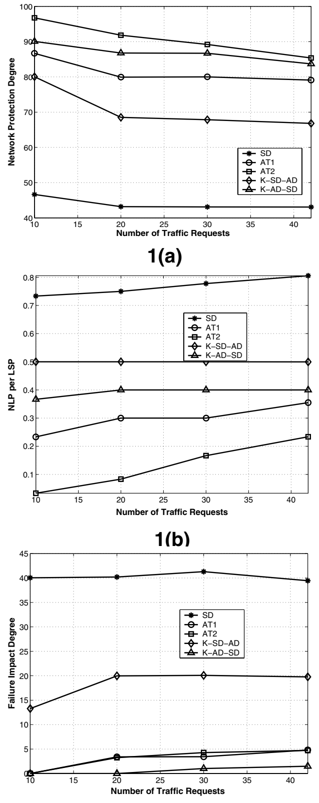
6.1. Simulation results

Simulation results are based on the network topology used in [3]. In the topology, 7 nodes are identified as traffic ingress and egress nodes (nodes 1, 2, 4,5, 9, 13 and 15). The capacity of the links is 1200 (normal lines) and 4800 (bold lines) units, and each link is bi-directional. The availability of each link is a pre-assigned value randomly generated between 0.9992 and 1. Each point in the simulation graphs is the average of 10 independent trials. Each trial uses an independent set of traffic flows. The bandwidth of each flow is uniformly distributed between 70 and 125 (unitless) and randomly assigned to ingress-egress node pairs. Note that, in our simulation we consider K=5 for *K-SD-AD* and *K-AD-SD*, since this value is adequate for this small network.

Figures 1(a)-(c) present the *NPD, FID* and *NLP* performance of the heuristic algorithm with the proposed cost functions as a function of total number of traffic flows. In these three figures we set the total percentage of low availability links to 30%. Figure 1(a) shows that *AT2* has the best *NPD*. This is due to the fact that for *AT2*, in most of the cases the costs associated to the high availability links are less than the low availability links. The other cost functions have lower *NPD* values but are still significantly better than *SD* which does not consider the link availability values at all and has the lowest *NPD* values. Figure 1(b) shows that *K-AD-SD* has the best *FID*. This is due to the fact that in *K-AD-SD*, penalizing the low availability links which are far away from the ingress router is considered directly as its first objective in the cost function. The other cost functions have higher *FID* values but are still significantly better than *SD,* which does not consider the failure impact at all. Figure 1(c) shows that *AT2* has the best *NLP* which means that fewer links needed to be protected and as a result less resources need be consumed by local backup paths for the same reasons as described for figure 1(a). Other cost functions have higher *NLP* values but are still significantly better than *SD*. Figures 1(a)-(c) show that among the four proposed cost functions *K-SD-AD* has the worse performance regarding *NPD,FID,NLP*. The reason is that this approach first finds *K* number of shortest paths according *SD* cost function and then as its second objective considers the availability and failure impact parameters to choose the final path. Hence, its availability performance is less than other proposed cost functions but still performs better than *SD*.

Figure 2(a) shows the primary path resource consumption of the proposed cost functions. The results show that in all of the proposed cost functions, the resources consumed by primary paths are more than those by *SD*. In fact, there is a trade off between the availability performance of the proposed cost functions and their primary path resource consumption. For example, since *AT2* has the best availability performance regarding *NPD* and *NLP*, it consumes 54% more resources than the *SD* on average. In comparison, *K-SD-AD* consumes only 6% more resources on average in comparison to that of *SD* since it provides the least availability among the four proposed cost functions. However, the resource consumption figures are markedly different when we include backup path resources, as shown in Figure 2(b). This shows that *SD* consumes the *most* resources in comparison to all the other proposed cost functions. This is due to the fact that most of its primary paths have low availability and backup paths need to be provisioned for them. However, our proposed cost functions require fewer backup paths to achieve overall path availability. Since our approach improves network path availability while reducing total network resource consumption, it can be an alternative to the existing approaches to achieve high availability and minimum cost network provisioning.

We summarise the performance of the proposed cost functions as follows. *AT1* has the best performance regarding total resource consumption. *AT2* has the best performance regarding *NPD* and *NLP*. *K-SD-AD* has the best performance regarding primary path *RC* after *SD*. *K-AD-SD* has the best performance regarding *FID*.



1(a)

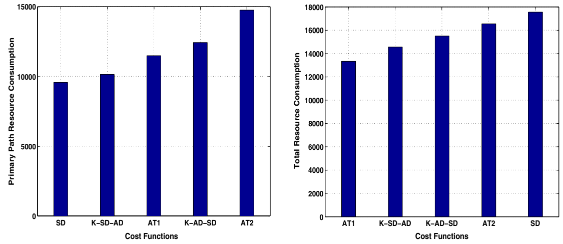
1(b)

1(c)

Figure 1. Effects of number of traffic flows on (a) NPD, (b) FID and (c) NLP

7. Conclusion

We have formulated the offline TE survivability design (OTESD) problem. The objective is to find for each traffic flow a primary path with improved availability and minimum failure impact that satisfies the bandwidth requirement while optimising resource consumption. We have proposed a heuristic algorithm with four various cost functions to solve the problem.



2(a)

2(b)

Figure 2. Resource consumption for (a) Primary paths and (b) Total paths

Simulation results show that our proposed heuristic algorithm increases the network protection, which means high availability primary paths can be provided for most of the traffic flows and so less failure detection, notification and traffic switching are required. Also, it decreases resource consumption for establishing local backup paths (if necessary). Finally, it can improve the resource efficiency by saving some amount of resources. In summary, this work allows ISPs to apply better TE and QoS routing strategies to increase service availability for their customers.

8. References

[1] C. Huang et al., “Building Reliable MPLS Networks using a path protection mechanism”, *IEEE Communication Magazine*, vol. 40, no. 3, March 2002.

[2] J.L. Marzo et al., “Adding QoS Protection in order to Enhance MPLS QoS Routing”, *IEEE International Conference on Communications*, vol. 3, 2003, pp. 1973-1977.

[3] E. Calle et al., “Protection Performance Components in MPLS Networks”, *Computer Communications Journal*, vol. 27, no. 12, July 2004, pp. 1220-1228.

[4] J. Zhang et al., “A New Provisioning Framework to Provide Availability-Guaranteed Service in WDM Mesh Networks”, *IEEE International Conference on Communications*, vol. 2, 2003, pp.1484-1488.

[5] Q. Ma et al., “On Path Selection for Traffic with Bandwidth Guarantees”, *IEEE International Conference on Network Protocols*, 1997, pp.191-202.

[6] D. Eppstein, “Finding the k Shortest Paths”, *35th IEEE Symposium on Foundations of Computer Science*, 1994, pp.154-165.

[7] A. Nucci et al., “IGP Link Weight Assignment for Transit Link Failures”*, Elsevier ITC18 2003,* Berlin, Germany.

[8] Telcordia document “Reliability Prediction Procedure for Electronic Equipment” (document number SR-332, Issue 1), AT&T Bell Labs 1999.

[9] M. To et al., “Unavailability analysis of long-haul networks”, *IEEE Journal on Selected Areas in Communications*, vol. 12, 1994, pp. 100-109.