Sharing Normal Bandwidth During a Failure

Rajarshi Gupta, *guptar@eecs.berkeley.edu*
Eric Chi, *echi@eecs.berkeley.edu*
Jean Walrand, *wlr@eecs.berkeley.edu*

Keywords: Protection Paths, Backup Paths, Quality of Service (QoS).

1. Introduction

We present a novel extension in network resource allocation that guarantees the protection of paths against an arbitrary failure of at most one link. It is a well known observation that the protection paths of disjoint normal paths are never active simultaneously and can consequently share resources [1, 2, 3, 4].

Our extension relies on the key observation that a failed normal path stops carrying traffic, and so remaining links on this path could potentially re-use this normal bandwidth to support other protection traffic. The chief contribution of our paper is the quantification of these additional gains by sharing reserved normal bandwidth that is diverted during a failure. Our scheme *complements* the above mentioned algorithms, rather than *competes* with them. We present simulation results that demonstrate non-trivial *additional* gains achieved by normal and backup sharing over backup sharing alone.

2. Centralized Algorithm

We first describe a centralized routing and resource accounting scheme that possesses complete information about the network, including the bandwidth allocated on each of the links. Let $B_N(i)$ and $B_B(i)$ denote the total normal and protection bandwidth allocated on link i. Let $B_T(i)$ be the total bandwidth on link i. We define $B_V(i)$, the available bandwidth as $B_V(i) = B_T(i) - B_N(i) - B_B(i)$. The demand for a new flow is represented by (s, d, b_N, b_B) where s is the source node, d the destination node and b_N and b_B the demanded normal and protection bandwidth respectively.

In addition to the link bandwidth values, the algorithm maintains the quantities $\{X(i,j), i \neq j \in E\}$ and $\{Y(i,j), i \neq j \in E\}$. For each $i \neq j \in E$, $X(i,j)$ is the amount of extra bandwidth that link i bears when link j fails. $Y(i,j)$ is the amount of normal bandwidth that goes away from link i when link j fails. Note that

$$B_B(i) = \max_j \{X(i,j) - Y(i,j)\}^+.$$

If link j fails, then link i carries the additional backup traffic $X(i,j)$ but carries $Y(i,j)$ less normal traffic.

Assuming that disjoint normal and backup paths have been determined through the network, we need to update the link state information accordingly, in order to handle future demands. Upon accepting a request (s, d, b_N, b_B) with normal path π_N and backup path π_B, we update the link states as follows:

$$\begin{aligned} X(i,j) &:= X(i,j) + b_B 1\{i \in \pi_B, j \in \pi_N\} \\ Y(i,j) &:= Y(i,j) + b_N 1\{i \in \pi_N, j \in \pi_N\} \end{aligned} \tag{1}$$

$B_N(i), B_B(i)$, and $B_V(i)$ are appropriately recomputed. When a flow (s, d, b_N, b_B) with normal path π_N and backup path π_B leaves, the reverse operations need to be carried out.

The centralized processor knows the resources available at each link. Thus, an on-line greedy algorithm selects a normal path (e.g. by executing Dijkstra's shortest path algorithm [5]) that satisfies the current capacity constraints. The algorithm then updates the link states as above, to generate a new intermediate set of capacity constraints. It then selects a backup path that meets these updated constraints. Finally, the algorithm updates the link states again to reflect the resources allocated for the backup path.

3. Distributed Algorithm

When conveying the state matrix from every link to the central processing unit is prohibitively burdensome, a distributed version of the algorithm is appropriate. In the distributed algorithm, every link i stores the same link state variables as above, but only the single value of $B_V(i)$ is distributed to every other node in the network.

When a node receives a demand (s, d, b_N, b_B), it executes its own QoS routing algorithm (e.g. Dijkstra) – first for the normal path and then for the protection path. While reserving the paths, the distributed algorithm requires that the list of links in the normal path be included in the resource reservation message for both the normal and backup paths. If a link i receives the normal path message, then it lies on the normal path and it re-calculates $Y(i,j) := Y(i,j) + b_N 1\{j \in \pi_N\}$. And if a link i receives the backup path message, then it needs to re-calculate $X(i,j) := X(i,j) + b_B 1\{j \in \pi_N\}$.

Although it is suboptimal, in many cases the distributed algorithm works as well as the centralized algorithm, except at the boundary condition as links saturate. Specifically, for a load-insensitive routing algorithm (e.g. minimum hop path), the distributed route calculations results in the same utilization of network resources until the last demand that saturates a link is reached.

4. Analysis

Let $X^*(i) = \max_k X(i,k)$ denote the greatest load of normal traffic that link i must bear in the event of a single link failure, without normal path sharing. Let $J^*(i) = \{j | X(i,j) = X^*(i)\}$ be the set of links that result in the worst case single link failure, without normal path sharing. And let $Y^*(i) = \min_{j \in J^*(i)} Y(i,j)$ be the minimum amount of normal flow common to link i and every link $j \in J^*(i)$.

$Y^*(i)$ and $X^*(i)$ together characterize the potential benefits of normal path bandwidth sharing. We define the gain for link i, $G(i)$ as the amount of backup resources that may be recovered by accounting for normal path sharing. We then derive the following upper bound,

$$G(i) \leq \min\{Y^*(i), X^*(i)\} \tag{2}$$

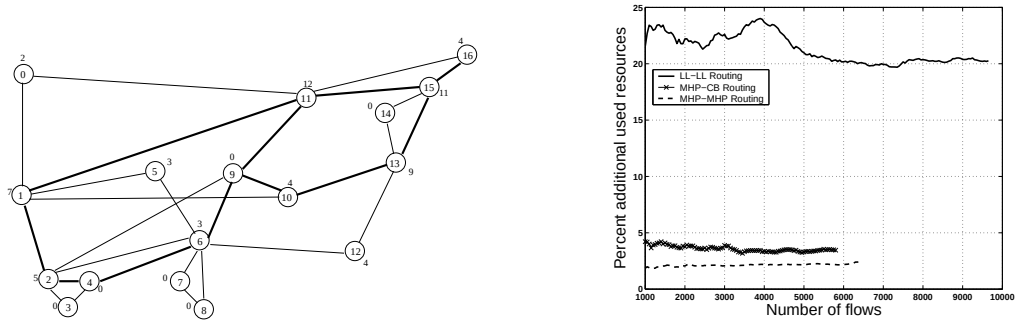
In the full paper, we also provide the general conditions when $G(i)$ achieves the upper bound.

5. Simulation Results

We present simulation results for a general USA and Metro Ring topology (for space constraints, only the USA Topology results are presented in this abstract). We simulate the arrival of connections that if admitted never depart. All connections require one unit of bandwidth for their normal path and one for their backup path. A realistic traffic load matrix is used. Simulation runs continue till the first blocked call.

We consider several common routing algorithms for normal and backup path computation: Least Loaded (LL), Minimum Hop Path (MHP), and Minimum Cumulative Backup (CB). LL routing chooses the route with the most spare capacity, MHP is the standard Dijkstra's shortest path algorithm, and CB chooses the path that minimizes the incremental increase in the total amount of protection resources reserved in the network. A variant of this scheme is employed in the AT&T IP Backbone Optical Network [6, 7] as shown in Figure 1(a). In the simulation results presented here, CB uses the centralized version of the algorithm while LL uses the distributed version. MHP is load independent, and so the two versions are equivalent.

Figure 1(b) plots the percent additional resources required when there is no normal path sharing – i.e. the ratio of backup path sharing alone versus backup and normal paths sharing. Plots start from 1000 admitted flows, to discount the relatively large initial fluctuations while plotting ratios. We observe differences of 2% for MHP-MHP, 4% for MHP-CB and up to 20% for LL-LL. In all cases, significant gains are observed,



(a) Topology (b) Percent Additional Used Resources

Figure 1: USA Simulation

with the most striking gains achieved while using LL-LL. This is a realistic instance of the topology, routing algorithm and traffic matrix together resulting in a situation where normal path sharing is highly beneficial, in line with the bounds presented in Section 4.

6. Conclusion

We improve on the methods used to calculate available network resources, independent of the QoS routing algorithms that are applied. We exploit the fact that when a link on a normal path fails, the traffic is diverted from this path and so remaining links on the normal path could potentially re-use this bandwidth to support other protection traffic. By maintaining normal path reservation state as well as backup path reservation state, worthwhile reductions in resource allocation may be achieved. The normal path sharing methods presented here should work in conjunction with any protection path sharing mechanisms, to augment the gains in resource allocation.

References

[1] Kodialam, M., and T. Lakshman, "Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration," *Proceedings IEEE INFOCOM 2000*.

[2] Liu, Y., D. Tipper and P. Siripongwutikorn, "Approximating Optimal Spare Capacity Allocation by Successive Survivable Routing," *Proceedings IEEE INFOCOM 2001*.

[3] Qiao, C., and D. Xu, "Distributed Partial Information Management (DPIM) Schemes for Survivable Networks - Part I," *Proceedings IEEE INFOCOM 2002*.

[4] Bouillet, E., J-F. Labourdette, G. Ellinas, R. Ramamurthy and S. Chaudhuri, "Stochastic Approaches to Compute Shared Mesh Restored Lightpaths in Optical Network Architectures," *Proceedings IEEE INFOCOM 2002*.

[5] Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, 1:269-271, 1959.

[6] UC Berkeley EECS 290Q Course website (Fall 2001): http://photonics.eecs.berkeley.edu/ee290q/jls/JLS%200911.pdf, Slide 37.

[7] AT&T CERFnet OC/12 Network Backbone website: http://www.cerf.net/about/Bbone-map.html