A Delay-Constrained Shared Mesh Restoration Scheme

Hassan Naser  
Department of Software Engineering  
Lakehead University  
Thunder Bay, Ontario, Canada  
hnaser@lakeheadu.ca

Ming Gong  
Faculty of Engineering  
Lakehead University  
Thunder Bay, Ontario, Canada  
mgong@lakeheadu.ca

Abstract—We present a multi-constrained routing algorithm, called Delay-constrained Pool Sharing (DPS), in a survivable mesh network. The goal of this algorithm is to find a pair of link-disjoint primary and backup paths between a given source and destination nodes, which guarantees full recovery from any single link failure in the network. Our objective is to minimize the resources (such as backup bandwidth) used in the network as well as the total end-to-end delay time along the primary and backup paths. Our approach improves the end-to-end delay performance of the Simple Pool Sharing (SPS) algorithm introduced in the literature, and proposes an integrated approach for Quality of Service (QoS) and resiliency. Using simulation, we studied the DPS algorithm on the existing North-American transport networks, and showed its improvements compared to the SPS scheme. We also introduce a hybrid algorithm which relaxes the delay constraint on the backup path in favor of maximizing the reusability (sharing) of the backup bandwidth.

Keywords-component; routing; quality of service; protection and restoration; network design and planning

I. INTRODUCTION

Current resiliency mechanisms focus on the restoration of network traffic in the event of a physical link (or node) failure in the network. Powerful dynamic protection and restoration algorithms have been developed for networks with different topology configurations, such as mesh or ring. Traditionally these algorithms have been applied at the physical layer (e.g. the optical layer), at which faults and alarms are detected.

On the other hand, Quality of Service (QoS) mechanisms focus on network capabilities that provide the facilities to differentiate network traffic and offer different levels of service to each class of traffic. Typical parameters that are used to measure QoS are: packet delay, packet delay variation, packet loss, etc. The QoS mechanisms have been mainly applied at layer 2 (e.g. ATM, MPLS) or layer 3 (IP) where packet forwarding and switching take place.

Recent technological breakthroughs can now facilitate novel forwarding techniques for optical data bursts that make it possible to capture packets at the optical layer. A major challenge in the transfer of these ultrahigh-speed data bursts is to allocate resources according to QoS specifications during the lifetime of data bursts (flows or connections) and to provide spare capacity required to address link failures in order to route these data bursts according to survivability requirements. Consequently, development of novel integrated strategies that facilitate implementation of QoS and survivability algorithms is of significant practical interest and is the primary focus of this study.

This paper is mainly concerned with improving the end-to-end packet delay in shared mesh restoration networks. Other QoS related parameters (such as packet loss, jitter) will be incorporated into our algorithms in the future. In a mesh restoration network, the source and destination nodes of the traffic traversing a failed link will reroute the traffic over one or a set of replacement paths (backup paths) between these nodes. In shared mesh restoration, the spare (backup) capacity is shared among different connections (traffic flows). The primary paths of these connections must be failure disjoint, so that no single failure can put out of service more than one connection at one time.

In this paper, we present three algorithms for computing a pair of link-disjoint primary and backup paths between a given source and destination nodes, in a shared mesh restoration network. First, we review a Simple Pool Sharing (SPS) mesh restoration algorithm introduced in our earlier contribution [2], and also elsewhere [3][4]. The objective of the SPS scheme is to minimize the primary and the spare capacities required for complete recovery of traffic from any single link failure. This is done by placing the spare capacity in each link into a common pool. Connections will share the resource pool in each backup link as long as their primary paths are failure disjoint. The SPS scheme is thus a single-constrained optimization scheme that computes a new pair of primary and backup paths in such a way that the total reserved working (primary) bandwidth and the total reserved backup bandwidth in the network are minimized.

We then introduce a new mesh restoration algorithm, called Delay-constrained Pool Sharing (DPS). The DPS scheme is a double-constrained optimization in that it attempts to reduce the end-to-end packet delay along the working and backup paths, and minimize the total reserved working and backup capacities in the network. While the DPS scheme may not achieve the optimal capacity performance of the SPS scheme in every network scenario, the DPS scheme does however yield a
lower end-to-end delay along both the working and backup paths compared with the SPS scheme.

The second new mesh restoration algorithm introduced in this paper is called Hybrid Pool Sharing (HPS). Like the DPS scheme, the HPS scheme takes into account the delay constraint when the working path is computed. However, this constraint is relaxed (not considered) when the backup path is computed, in order to maximize the reusability (sharing) of the backup bandwidth. Therefore, in terms of the capacity performance, we will show that the HPS scheme performs in the middle for all the presented schemes. In terms of the delay along the working paths, it performs as close as the DPS scheme. And, in terms of the delay along the backup paths, it performs as close as the SPS scheme.

The three routing schemes discussed in this paper can be used in QoS-based survivable packet networks with three general types of services. The SPS scheme is suitable to mainstream network applications that may require reliable delivery of data, but are not sensitive to network latency (delay). The DPS scheme is suitable to mission critical applications that require reliable delivery of data and are also delay-sensitive. Finally, the HPS scheme is suitable to applications that require reliable delivery, and will enjoy small network delay during the normal operation of the network. However, these applications can tolerate increased delay when there is a fault in the network.

The reminder of the paper is structured as follows. Section II gives an overview of the background restoration techniques in mesh networks. Section III reviews Simple Pool Sharing (SPS) restoration scheme, and Section IV presents Delay-constrained Pool Sharing (DPS) scheme. Section V presents three representative North American test networks, and demonstrates the packet delay and capacity performances of the SPS and DPS schemes implemented in these networks. Section VI presents the Hybrid Pool Sharing (HPS) scheme, and compares its delay and capacity performances with the SPS and DPS schemes. Section VII summarizes this paper.

II. BACKGROUND

Many studies have explored the capacity saving of various shared mesh restoration schemes over the alternative protection and restoration schemes. Less studies has however been carried out on the evaluation of the Quality of Service (QoS) performance of the shared mesh restoration schemes. One example is a restoration scheme proposed in [9] which attempts to quantify QoS in WDM shared mesh restoration networks. However, in their paper, QoS represented the amount of time that a connection is down, rather than the packet-level QoS parameters (such as delay, jitter, or loss).

Another scheme is a Simple Pool Sharing (SPS) scheme originally introduced in [2]. The SPS scheme is a two-step algorithm that finds a shortest working path in the first step and a least-capacity backup path for that working path in the second step. We will review this scheme in the next section. In [6], a linear programming formulation for shared path protection is introduced so that the least-cost link-disjoint working and protection paths can be derived in a single step. In [7] and [8], two heuristic routing algorithms for shared protection are discussed. In all of the papers referenced above, “blocking probability” and “capacity usage” have been used as the criteria for path selection. None of these schemes have taken QoS related parameters into consideration.

Reference [10] gives a tutorial on QoS routing and reviews techniques and algorithms introduced in the literature for satisfying QoS requirements for every admitted connection. One of such algorithms is the algorithm presented in [11] which proposes a heuristic algorithm for an NP-complete delay-constrained least-cost routing problem. The goal of this algorithm is to find a path that has the highest probability to satisfy a given end-to-end packet delay bound. Another QoS routing algorithm is the one in [12] which studies a bandwidth-constrained and delay-constrained routing problem with imprecise network states. Yet, another algorithm is the one presented in [13] which finds a bandwidth-delay-constrained path by Dijkstra shortest path algorithm. The above three algorithms are only the examples of many other QoS routing algorithms developed in recent years. However, almost none of these algorithms have taken protection and restoration parameters into consideration.

III. SIMPLE POOL SHARING SCHEME

The Simple Pool Sharing (SPS) scheme is a sub-optimal scheme, in the sense that it computes a pair of working and backup paths for every given source-destination nodes in two steps [2]. In step 1, the working path is computed, whereas in step 2 a link-disjoint backup path is computed. The SPS scheme places the backup bandwidth in each link into a common pool. It then ensures that the amount of backup bandwidth required on a backup link to restore failed connections (resulting from any single link failure) will not be more than the total amount of backup bandwidth reserved on that link. The pool sharing process records the backup bandwidth reserved on links in the following matrix [2]:

\[
K = \begin{bmatrix}
0 & k_{12} & k_{13} & \ldots & k_{1J} \\
k_{21} & 0 & k_{23} & \ldots & k_{2J} \\
k_{31} & k_{32} & 0 & \ldots & k_{3J} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
k_{J1} & k_{J2} & k_{J3} & \ldots & 0
\end{bmatrix}
\]

(1)

Element \(k_{ij}\) is the amount of backup bandwidth needed on link \(j\) if link \(i\) fails (\(1 \leq i, j \leq J\)), where \(J\) is the number of links in the network. In pool sharing, the total amount of backup bandwidth needed on link \(j\) \((B_j)\) is indeed the maximum of all elements in column \(j\), which is:

\[
B_j = \max_{1 \leq i \leq J} \{k_{ij}\}
\]

(2)

Let us define the following additional notations: \(A_i\) denotes the available bandwidth on link \(i\); \(C^w_i\) denotes the cost of link \(i\) for working path computation; \(C^b_j\) denotes the cost of link \(j\) for backup path computation; and \(b\) denotes the amount of bandwidth requested by a newly arrived demand \(r\).
The SPS scheme uses the following 2-step procedure to find a pair of paths between the source and destination nodes of the newly arrived demand \( r \). First it computes a working path with the least capacity among all available paths between these two nodes. The cost of link \( i \) is defined as the following [2]:

\[
C^w_i = \begin{cases} 
\infty & b > A_i \\
b & b \leq A_i 
\end{cases} \quad \forall i \in [0,...,J-1] 
\]

(3)

The cost of link \( i \) is set to \( b \) if the link has enough available capacity to allocate the bandwidth requested by the demand \( r \). Otherwise, the cost is set to an arbitrary large number, denoted as \( \infty \). Dijkstra algorithm [1] is run to compute the shortest (min-hop) working path for demand \( r \) using the above cost assignment. We denote the set of links along the computed working path as \( S_w(r) \).

In the second step, the SPS scheme computes a backup path by executing Dijkstra algorithm again, but this time the cost of every link \( j \) is set to \( C^b_j \) defined below. Let \( T_j \) denote the maximum amount of backup bandwidth required on link \( j \) if a link in \( S_w(r) \) fails. It follows that \( T_j \) will simply be:

\[
T_j = b + \max_{i \in S_w(r)} \{ k_i \} 
\]

(4)

The cost function \( C^b_j \) is then defined as the following [2]:

\[
C^b_j = \begin{cases} 
\infty & j \in S_w(r) \\
\varepsilon & T_j \leq B_j \\
T_j - B_j & 0 < T_j - B_j \leq A_j \\
\infty & \text{otherwise}.
\end{cases}
\]

(5)

The cost of link \( j \) is set to \( \infty \) if \( j \) is along the working path of demand \( r \). The cost is set to a very small number \( \varepsilon \) if \( j \) is not in \( S_w(r) \) and if \( T_j \) is less than \( B_j \). In this case, demand \( r \) can be restored on link \( j \) without need to reserve any additional backup bandwidth on this link. In other words, this demand shares the backup capacity on link \( j \) with other demand(s). If neither of the above conditions is satisfied, the cost is set to \( (T_j - B_j) \) if this quantity is less than the available capacity on link \( j \). In this case, \( (T_j - B_j) \) is the amount of additional backup bandwidth required on link \( j \) in order to restore demand \( r \). If the available capacity on link \( j \) is not enough to accommodate this additional bandwidth, the cost of link \( j \) is set to \( \infty \).

If the backup path is found it will be an optimal path (with the least required backup capacity) that is also link-disjoint from the path computed in step 1. Once the backup path is computed, the total reserved shared backup bandwidth on links along the backup path must be updated by updating the elements of matrix \( K \) (as described in [2]). We denote \( S_b(r) \) as the set of links along the backup path of demand \( r \).

IV. DELAY-CONSTRAINED POOL SHARING SCHEME

The Delay-constraint Pool Sharing (DPS) scheme introduced in this paper is also a sub-optimal scheme, which computes a pair of working and backup paths for a demand \( r \) in two steps. The DPS scheme however uses different link cost assignments than those used by the SPS scheme to determine the paths.

In the first step of the algorithm, the DPS process computes a working path for demand \( r \) that (a) satisfies the capacity requirement of this demand and, (b) achieves a minimum end-to-end packet delay. Let \( d_i \) represent the average delay that the previously transmitted packets experienced on link \( i \). In general, \( d_i \) can have three components: queuing delay, transmission delay, and propagation delay. The cost of link \( i \) is then defined as the following:

\[
C^w_i = \begin{cases} 
\infty & b > A_i \\
d_i & b \leq A_i 
\end{cases} \quad \forall i \in [0,...,J-1] 
\]

(6)

The cost of link \( i \) is set to \( d_i \) if the link has enough available capacity for demand \( r \). Otherwise, the cost is set to \( \infty \). The above cost assignment is used by Dijkstra algorithm to find a least-delay working path for demand \( r \).

In the second step, the DPS scheme computes a least-delay link-disjoint shared backup path for the working path. Demand \( r \) shares the backup bandwidth on links along the backup path with other demands. Following the notations used in the previous section, we denote \( S_b(r) \) to be the set of links along the working path of demand \( r \), and \( T_j \) to be the maximum amount of backup bandwidth required on link \( j \) if a link in \( S_b(r) \) fails. \( T_j \) is computed according to (4). \( C^b_j \) is then defined as the following:

\[
C^b_j = \begin{cases} 
\infty & j \in S_b(r) \\
d_j & T_j \leq B_j \\
d_j + \delta & 0 < T_j - B_j \leq A_j \\
\infty & \text{otherwise}.
\end{cases}
\]

(7)

The above cost assignment has two main differences from the cost assignment (5). It sets the cost of link \( j \) to \( d_j \) (rather than to \( \varepsilon \)) if on this link adequate shared backup bandwidth had already been reserved to restore demand \( r \) (i.e. \( T_j \leq B_j \)). Therefore, if more than one backup link meet the above bandwidth condition, the one with the lowest delay is favored. The cost function (7) also differs from (5) in that it sets the cost of link \( j \) to \( d_j + \delta \) (rather than to \( T_j - B_j \)) if this link satisfies the third condition. We remark that a link \( j \) meets this condition if \( T_j - B_j \) units of additional backup bandwidth must be reserved on this link in order to restore demand \( r \), and if this bandwidth can be honored (if \( 0 < T_j - B_j \leq A_j \)). Parameter \( \delta \) is a nonnegative number which is added to allow the path computation algorithm to favor a link that meets condition 2 over a link that meets condition 3, if the delays on both links are equal. This ensures that the DPS scheme achieves a least-delay backup
path while still maximizing backup bandwidth sharing. Our findings in this paper will show that even if $\delta$ is set to zero, there would still be a significant backup bandwidth sharing.

V. SIMULATION RESULTS

In this section we compare the performances of the SPS and DPS schemes. Our criteria for performance evaluation are: 1) the sum of the working bandwidth reserved on all links in the network, 2) the sum of the backup bandwidth reserved on all links in the network, 3) the end-to-end delay along the working path averaged over the number of accepted demands (denoted as $D_w$), and 4) the average end-to-end delay along the backup path ($D_b$). If we denote the number of demands accepted to the network by $N$, the above two parameters can be computed as:

$$D_w = \frac{1}{N} \sum_{r=1}^{N} \left( \sum_{i \in S_p(r)} d_i \right), \quad D_b = \frac{1}{N} \sum_{r=1}^{N} \left( \sum_{i \in S_b(r)} d_i \right) \quad (8)$$

We used three networks (NSF, GCN and MCI) and four demand matrices for each network to test the criteria identified earlier. These networks are representative of the North American backbone networks. NSF is based on the National Science Foundation network, GCN is based on the Global Crossing network, and MCI is based on the MCI IP network. The topologies of these networks are illustrated in Fig. 1, and their important characteristics are shown in Table 1.

In this paper, we assume that all links in the network have infinite capacity so that the demands will never be rejected due to insufficient capacity. We generated four demand matrices for each network. Three of them are random demand matrices, with the total number of demands in each matrix being 100, 200 and 500, respectively. For every demand in these matrices, the source and destination nodes are generated randomly according to uniform distribution between 1 to $M$, where $M$ is the number of nodes in the network. The fourth demand matrix is a distributed traffic demand matrix [14], in which between every pair of nodes in the network a number of point-to-point demands exist. This number is a function of the population of these nodes (which represent popular North American cities) and the overall population of all nodes in the network. In this case, the number of generated demands is 240 for the NSF network, 702 for the GCN, and 1406 for the MCI network. In our simulations, all demands requested identical amount of bandwidth from the network; parameter $b$ was set to 10 Mbps for all demands. When DPS scheme is used, delay $d_i$ is set to the propagation delay on link $i$, and parameter $\delta$ is set to zero.

The related packet delay and capacity performances of the SPS and DPS schemes in the NSF network are shown in Figs. 2 and 3, each as a function of the number of demands accepted in the network. Fig. 2 shows the total reserved working capacity for each scheme (denoted in the figure as $SPS_w$ and $DPS_w$), the total reserved backup capacity ($SPS_b$ and $DPS_b$), and the sum of these two quantities for each scheme ($SPS_{tot}$ and $DPS_{tot}$).

As shown in Fig. 2, when the SPS scheme is used, the above capacity quantities are smaller than their corresponding quantities for the DPS scheme. This is intuitive, because the SPS scheme attempts solely to minimize the capacity usage in the network. Whereas the DPS scheme concentrates on minimizing end-to-end delay; the capacity usage is of secondary importance. For this reason, the average end-to-end delay along a working path ($D_w$) and along a backup path ($D_b$) for the DPS scheme are lower than the corresponding parameters for the SPS scheme (see Fig. 3). Similar observations can be made when we study the capacity and delay performances of the SPS and DPS schemes in GCN and MCI networks (Figs. 4 through 7).
One important observation that can be made from Figs. 2 through 7 is that in all of the three tested networks, when the network is lightly loaded (i.e. 100 demands in the network) the capacity performances of the DPS scheme approach to their corresponding quantities of the SPS scheme. However, this rule does not apply when delay parameters are considered. In Figs. 3, 5, and 7, the DPS scheme always outperforms the SPS scheme in terms of the working and backup delays, and the difference in the delay performances of the two schemes is almost fixed at all load levels studied. Therefore, it appears from this discussion that the DPS scheme can potentially achieve the optimal capacity performance of the SPS scheme, while still outperforming it in terms of the packet delay at low network loads.

VI. HYBRID POOL SHARING SCHEME

The SPS scheme can be used to find the least-capacity shared restoration solution for many internet applications that are not sensitive to network delay latency time. In contrast, the DPS scheme can be used to find the least-delay shared restoration solution for delay-sensitive internet applications. However, there may exist some internet applications (e.g. interactive applications such as Web call-center) that want to receive low delay service when the network is normal, but can tolerate increased delay when the network is experiencing failure. During this time, connectivity is all that these applications want from the network no matter how much delay they would experience. For these applications, the SPS scheme may not provide adequately low working delay, and the DPS scheme may be costly because of its relatively high backup capacity usage. In this section, we introduce an alternative scheme, called Hybrid Pool Sharing (HPS), which can yield a low working path delay and at the same time it can save some backup capacity.

The HPS scheme uses the DPS scheme’s working path cost function (6) to compute a low-delay working path, but it uses the SPS scheme’s backup path cost function (5) to compute a least-capacity shared backup path. Figs. 8, 9, and 10 compare the total reserved capacity (the sum of the working and backup capacities), the average end-to-end delay along a working path, and the average end-to-end delay along a backup path of the three schemes (SPS, DPS, HPS) in the NSF network. As it is shown in Fig. 8, the HPS scheme consumes more capacity than the SPS scheme, because its working paths are less capacity optimized. It consumes less capacity than the DPS scheme, because it maximizes backup bandwidth sharing among backup paths.

In Fig. 9, the working delay of the HPS scheme is very close, almost identical, to the working delay of the DPS scheme. Whereas, in Fig. 10, the HPS backup delay is the same as the backup delay of the SPS scheme. Similar observations were made when we compared these three schemes on the GCN and MCI networks. The corresponding results have been omitted for the sake of space.
VII. CONCLUSION

We have developed a double-constrained shared mesh restoration algorithm, called Delay-constrained Pool Sharing (DPS), which computes a pair of link-disjoint paths between any given source and destination nodes with consistently low latency (delay) on an end-to-end basis. The algorithm follows from our earlier published algorithm, called Simple Pool Sharing (SPS), except, in this case, two routing constraints (delay and capacity) are optimized rather than one (the capacity). For 5-to-20% increase in shared backup capacity, the DPS scheme yields 16-to-34% reduction in the end-to-end delay along the backup path. It also yields 7-to-15% reduction in the delay along the working path when it is compared with the SPS scheme. We have also developed a hybrid algorithm called Hybrid Pool Sharing (HPS) intended to combine the merits of the DPS and SPS schemes. While the HPS scheme achieves the same end-to-end delay bound along the working path as DPS, it yields the same amount of backup bandwidth sharing as SPS.

REFERENCES