NPP: A Facility Based Computation Framework for Restoration Routing Using Aggregate Link Usage Information*

Faisal Aslam, Saqib Raza, Fahad Rafique Dogar,
Irfan Uddin Ahmad, and Zartash Afzal Uzmi

Lahore University of Management Sciences, Pakistan
{faisal,saqibr,fahad,irfank,zartash}@lums.edu.pk

Abstract. We present NPP – a new framework for online routing of bandwidth guaranteed paths with local restoration. NPP relies on the propagation of only aggregate link usage information [2,9] through routing protocols. The key advantage of NPP is that it delivers the bandwidth sharing performance achieved by propagating complete per path link usage information [9], while incurring significantly reduced routing protocol overhead. We specify precise implementation models for the restoration routing frameworks presented in [1] and [2] and compare their traffic placement characteristics with those of NPP. Simulation results show that NPP performs significantly better in terms of number of LSPs accepted and total bandwidth placed on the network. For 1000 randomly selected LSP requests on a 20-node homogenous ISP network [8], NPP accepts 775 requests on average compared to 573 requests accepted by the framework of [2] and 693 requests accepted by the framework of [1]. Experiments with different sets of LSP requests and on other networks indicate that NPP results in similar performance gains.

1 Introduction

The destination based forwarding paradigm employed in plain IP routing does not support routing along explicit routes determined through constraint based routing [12]. The emergence of Multi-Protocol Label Switching (MPLS) has overcome this limitation of traditional shortest path routing, by presenting the ability to establish a virtual connection between two points on an IP network, maintaining the flexibility and simplicity of an IP network while exploiting the ATM-like advantage of a connection-oriented network [13]. Ingress routers of an MPLS network classify packets into forwarding equivalence classes and encapsulate them with labels before subsequently forwarding them along pre-computed paths [15]. The path a packet takes as a result of a series of label switch operations in an MPLS network is called a label switched path (LSP). LSPs may be routed through constraint based routing that adapts to current network state information (e.g., link utilization) and selects explicit routes that satisfy a set of

* This work was supported by a research grant from Cisco Systems, San Jose, CA.

M. Ajmone Marsan et al. (Eds.): QoS-IP 2005, LNCS 3375, pp. 150–163, 2005.

constraints. The ability to explicitly route network traffic using constraint based routing enables service providers to provision quality of service and also leads to efficient network utilization [14].

An important application of constraint based routing is provisioning of bandwidth guaranteed LSPs [6–8]. Furthermore, many real-time applications require that the guaranteed bandwidth remains available when network facilities[1] fail. When recovery mechanisms are employed at the IP layer, restoration may take several seconds which is unacceptable for real-time applications [11]. In contrast, MPLS *local restoration* meets the requirements of real-time applications with recovery times comparable to those of SONET rings [6,10]. In local restoration, each LSP passing through a facility is protected by a backup path which originates at the node immediately upstream to the facility. This node, which redirects the traffic onto the preset backup path in case of failure, is called the Point of Local Repair (PLR). Since this decision to redirect traffic is strictly local, faster recovery is possible. In this paper, we consider routing bandwidth guaranteed paths with local restoration.

There are two distinct approaches to local restoration: In *one-to-one* backup approach [6–8], the PLRs maintain separate backup paths for each LSP passing through a facility. The backup path terminates by merging back with the primary path at a node called the Merge Point (MP). In one-to-one backup approach, the MP can be any node downstream the protected facility. Maintaining state information for backup paths protecting individual LSPs, as in the one-to-one approach, is a significant resource burden for the PLR. Moreover, periodic refresh messages[2] sent by the PLR, in order to maintain each backup path, may become a network bottleneck [4]. On the other hand, in *many-to-one* approach, a PLR maintains a single backup path to protect a set of primary LSPs traversing the triplet (PLR, facility, MP)[3]. Thus, fewer states need to be maintained and refreshed which results in a scalable solution. Many-to-one backup approach, also called *facility backup*, is illustrated in Fig. 1. Note that in this approach, the MP should be the node immediately downstream to the facility.

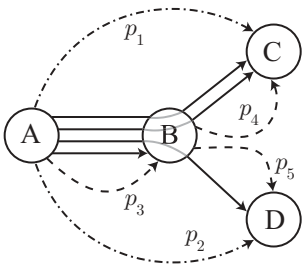
Backup provisioning requires bandwidth reservation along the backup paths, thereby reducing the total number of LSPs that can otherwise be placed on the network. This reduction is significant if resources along the backup paths are not shared. Since it is reasonable to assume that different network facilities will not fail simultaneously [1, 6–8], backup paths protecting different facilities should share bandwidth.

In this paper, we present NPP – a new framework for facility backup, which relies on the propagation of aggregate link usage information through routing protocols [2, 17]. The key advantage of NPP is that it delivers the bandwidth sharing performance achieved by propagating complete per path link usage in-

[1] The term facility refers to either a node or a bidirectional link.

[2] Local Protection primarily uses RSVP-TE extensions, which is a soft-state protocol and requires periodic refresh messages to maintain its states.

[3] Throughout this paper, an LSP traversing the triplet (PLR, Facility, MP) refers to a primary LSP that passes through the PLR, the facility, and the MP in that order.



Backup Path	Protects All LSPs Traversing
p_1	A-B-C
p_2	A-B-D
p_3	A-B
p_4	B-C
p_5	B-D

Fig. 1. Many-to-one or Facility Backup.

formation [9], while incurring significantly reduced routing protocol overhead. We compare the traffic placement characteristics of NPP with two existing facility backup frameworks, described in [1] and [2], and present results based on the total number of LSP demands accepted and the bandwidth placed on the network.

2 Background

2.1 Routing Paradigm

The optimality of primary and backup paths computed to serve an LSP request is a function of network state information available during path computation. Two path computation scenarios are possible in this context: Centralized and Distributed. In centralized path computation, one central entity makes all the routing decisions. This central entity can achieve optimal backup bandwidth sharing as it maintains complete network information. However, a centralized path computation server incurs additional costs in terms of high processing power and high bandwidth control channels [5]. Distributed control entails autonomous path computation by distributed nodes, based on the node's view of the network state. The state maintained by a node, in the distributed path computation scenario, is a function of its local state and the network state information periodically distributed by other routers via link state routing protocols. We consider restoration routing frameworks that use distributed path computation, such that the primary path is computed by the LSP ingress node and backup paths are computed locally. The primary and backup paths are signaled using protocols like RSVP-TE [3] and CR-LDP [18]. In case of primary paths, the ingress node initiates the signaling, while each backup path is signaled by its PLR. Each node along the signaled route reserves the required bandwidth, before forwarding the signaling request to the next node along the path.

2.2 Fault Model

Failure of a network facility along an LSP results in the need to divert the LSP traffic onto a preset backup path. Failure of network nodes and links is a low probability event, and network measurements reveal that chances of multiple

failures in the network are even lower. Furthermore, upon failure along the primary path new reoptimized primary and backup paths may be provisioned, with local restoration serving only as a temporary measure [1]. The probability of additional failures during the setup of reoptimized paths is negligible. Therefore, a more realistic restoration objective is to provide protection against the failure of a single facility. We consider the fault model wherein backup paths provision restoration in the event of a single link or single node failure. We refer to this fault model as the *single element protection* fault model.

In order to elucidate local recovery for the single element protection fault model we distinguish between two types of backup paths: *next-hop* paths and *next-next-hop* paths.

Definition 1. *A next-hop path that spans a link* (i, j)[4] *is a backup path which:*

a) originates at node i*,*
b) merges with the primary LSP(s) at node j*, and*
c) provides restoration for one or more primary LSPs that traverse (i, j)*, if* $\{i, j\}$ *fails.*

Definition 2. *A next-next-hop path that spans a link* (i, j) *and a link* (j, k) *is a backup path which:*

a) originates at node i*,*
b) merges with the primary LSP(s) at node k*, and*
c) provides restoration for one or more primary LSPs that traverse (i, j)*, if* $\{i, j\}$ *or node* j *fails.*

Fig. 2 depicts local restoration with respect to a single primary path according to the single element protection fault model. The figure shows that setting up next-next-hop paths along the primary path provides restoration in event of single node failure. Note that such a configuration also protects against the failure of all except the last link. In order to provision single element protection, an additional next-hop backup path spanning the last link is setup.

3 Problem Definition

We consider a network with n nodes and m bidirectional links. LSP requests arrive one by one at the ingress node, and the routing algorithm has no a priori knowledge of future requests. An LSP request is characterized by the LSP ingress node, the LSP egress node, and an associated bandwidth demand b.

In order to serve an LSP request, a bandwidth guaranteed primary path must be setup along with locally restorable backup paths that provide protection against the failure of facilities along the primary path. If the routing algorithm

[4] A bidirectional link between two nodes constitutes a single facility. However, traffic traverses a link in a specific direction. We, therefore, use $\{i, j\}$ to represent the bidirectional link between node i and node j, and use the ordered pair (i, j) when direction is significant. Thus, (i, j) refers to the directed stem of $\{i, j\}$ from node i to node j. Note that failure of the facility $\{i, j\}$ implies failure of (i, j) and (j, i).

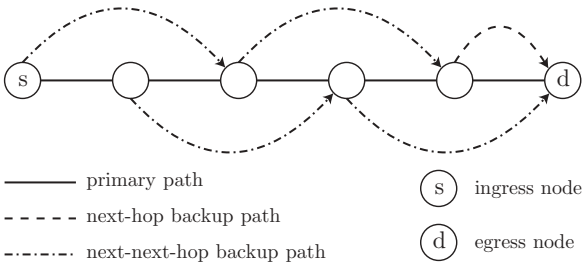


Fig. 2. Local Restoration for Single Element Failure.

is able to find sufficient bandwidth in the network for the requisite primary and backup paths, the paths are setup, and the LSP request is accepted; otherwise, the LSP request is rejected. The next LSP request arrives only after the current LSP request has either been accepted or rejected. Our goal is to optimize network utilization. To this end, we wish to minimize the bandwidth reserved for the primary and backup paths for each LSP request.

Suppose we are serving a new LSP request with bandwidth demand b. Further suppose that the computed primary LSP for this request traverses (i, j). A backup path, that provisions restoration for this primary LSP along (i, j), is a next-hop path if j is the egress node, and is a next-next-hop path otherwise. Each backup path protects one or more facilities and has a merge point. The PLR for both types of backup paths is node i. In sum, we are trying to protect a primary LSP that traverses the triplet (PLR, facility, MP). It is possible that a backup path $\wp_{\text{old}}$ already exists, that protects previously routed primary LSPs traversing the same triplet. If such a backup path exists, restoration for the new request may be provisioned by reserving additional bandwidth along that path. However, some links along $\wp_{\text{old}}$ may not have the capacity for the requisite additional reservation. Therefore, if the primary LSP belonging to the current LSP request traverses the triplet (PLR, facility, MP), the PLR re-computes a new backup path that provisions restoration for all primary LSPs traversing that triplet, including the new primary LSP. We refer to this new backup path as $\wp$. If the cumulative bandwidth reserved for the previous LSPs traversing the triplet (PLR, facility, MP) was b_{old}, the bandwidth required for $\wp$ is given by $b_{\text{new}} = b + b_{\text{old}}$. The details for path computation, bandwidth sharing, and subsequent setup of $\wp$ depend upon the restoration routing framework and are explained in the following sections.

4 Backup Bandwidth Sharing

Failure of a protected facility (link or node) results in the activation of a set of backup paths. We refer to the set of backup paths that are simultaneously activated, if a facility fails, as the *activation set* for that facility. PLRs detect[5]

[5] Mechanisms exist that allow the PLR to distinguish between link and node failure; see [16] for details.

link or node failure and subsequently activate all backup paths that protect the failed facility. The following enumerates the backup paths included in the activation sets for the facilities $\{i, j\}$ and node j:

Activation Set for $\{i, j\}$: Recall from section 2 that a next-hop path protects against failure of a link, and a next-next-hop path protects against failure of both a link and a node. Incase $\{i, j\}$ fails all next-hop and next-next-hop paths protecting $\{i, j\}$ are activated. This activation set for $\{i, j\}$ comprises the following backup paths (see Fig. 3):

- next-hop path that spans (i, j)
- next-hop path that spans (j, i)
- next-next-hop paths that span (i, j) and $(j, x), \forall x \neq i$
- next-next-hop paths that span (j, i) and $(i, x), \forall x \neq j$

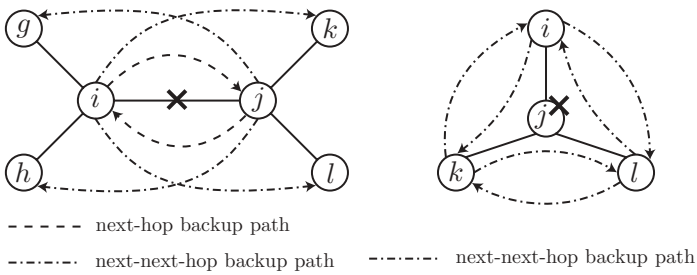


Fig. 3. Paths in the Activation Set for link $\{i, j\}$ (left) and for node j (right).

Activation Set for Node j: Recall from section 2, only next-next-hop paths protect against failure of a node. Thus, the activation set for node j comprises the next-next-hop paths that span (x, j) and $(j, y), \forall x \neq y$. Fig. 3 shows the set of backup paths that are activated incase node j fails.

Since only a single link or a single node may fail at a time, two backup paths will not be simultaneously active unless they are in the same activation set. It follows that such backup paths can share bandwidth with each other. In contrast, backup paths that are simultaneously active must make bandwidth reservations that are exclusive of each other. Consequently, a next-hop path that spans (i, j) is activated if $\{i, j\}$ fails, and therefore, cannot share bandwidth with other backup paths belonging to the activation set of $\{i, j\}$. Similarly, a next-next-hop path that spans (i, j) and (j, k) is activated if either $\{i, j\}$ or node j fails, and hence cannot share bandwidth with backup paths belonging to the activation sets of either $\{i, j\}$ or node j.

5 Restoration Routing Frameworks

The extent of bandwidth sharing is governed by a number of parameters that include the distribution of path computation, the amount of network state information propagated through routing protocols, and the signaling mechanisms

used for path setup. In this section, we present two existing restoration routing frameworks. We then introduce NPP, our own restoration routing framework, and describe how it draws upon the advantages of the other two frameworks to achieve more efficient bandwidth sharing.

5.1 Kini's Framework

In this framework, each backup path is computed by its PLR. The PLR relies on network state information propagated through routing protocols to decide how much bandwidth a backup path can share on any given link. Kini's framework involves propagation of aggregated per-link network usage information as proposed in [2]. Kini's framework is characterized by two distinct stages: a suboptimal path computation stage, and a corrective signaling stage.

Suboptimal Path Computation Stage: In the suboptimal path computation stage, the PLR computes a backup path, using aggregate link usage information, to make bandwidth sharing decisions. Maximum sharing between backup paths can be achieved if per-path information is available at the path computation server [9]. However, making such information available incurs significant protocol overhead in terms of network utilization, memory, update-processing and associated context switching [14]. Kini, et al. have shown in [2] that propagation of aggregated per-link network usage information can result in cost-effective sharing between backup paths.

Aggregated per-link network usage information involves propagation of the following values for each link (i, j) in the network:

$\boldsymbol{F_{ij}}$ **:** Bandwidth reserved on (i, j) for primary LSPs
$\boldsymbol{G_{ij}}$ **:** Bandwidth reserved on (i, j) for backup LSPs
$\boldsymbol{R_{ij}}$ **:** Residual bandwidth on (i, j)

The above network state information (i.e., F_{ij}, G_{ij}, and R_{ij}) can be propagated using traffic engineering extensions to existing link state routing protocols [17]. The following describes how Kini's framework makes use of this aggregate link usage information to make bandwidth sharing decisions.

In order to route $\wp$, we seek to find the amount of bandwidth reserved on a link (u, v) that can be shared by $\wp$, for every (u, v) in the network. To this end, we consider the following two cases:

Case 1: $\wp$ Is a Next-Hop Path That Spans a Link (i, j). Recall from section 2 that a next-hop path that spans (i, j) is activated when $\{i, j\}$ fails. As described in section 4, the activation set of $\{i, j\}$ consists of next-hop and next-next-hop backup paths that provision restoration for primary LSPs that traverse $\{i, j\}$. The total amount of bandwidth for these LSPs is $F_{ij} + F_{ji}$. Thus, a link (u, v) can have a maximum of $F_{ij} + F_{ji}$ units of backup bandwidth reserved for backup paths that are activated when $\{i, j\}$ fails. However, b_{old} units of bandwidth, out of $F_{ij} + F_{ji}$, are reserved for $\wp_{\text{old}}$. Therefore, the amount of bandwidth that is simultaneously active with $\wp$ is given by $F_{ij} + F_{ji} - b_{\text{old}}$. It follows that S_{uv}, the bandwidth available for sharing by $\wp$ on (u, v), is given by:

$$S_{uv} = \max(0, G_{uv} - F_{ij} - F_{ji} + b_{\text{old}}) \tag{1}$$

Case 2: $\wp$ Is a Next-Next-Hop Path That Spans a Link (i, j) and a Link (j, k). Recall from section 4 that a next-next-hop path that spans (i, j) and (j, k) is activated if either $\{i, j\}$ or node j fails. As explained in Case 1, $F_{ij} + F_{ji} - b_{\text{old}}$ is the worst case bandwidth reserved for backup paths that will be simultaneously active with $\wp$ when $\{i, j\}$ fails. Similarly, the activation set of node j comprises next-next-hop paths that protect primary LSP traffic traversing all links (x, j), such that node x is adjacent to node j. The maximum amount of such traffic is given by $\sum_x F_{xj}$. Thus, a link (u, v) can have a maximum of $\sum_x F_{xj}$ units of backup bandwidth reserved for backup paths that are activated when node j fails. However, b_{old} units of bandwidth, out of $\sum_x F_{xj}$, are reserved for $\wp_{\text{old}}$. Therefore, the maximum amount of bandwidth that is simultaneously active with $\wp$, when node j fails, is given by $\sum_x F_{xj} - b_{\text{old}}$. Since $\wp$ is activated if either node j or $\{i, j\}$ fails, a link (u, v) can have up to $\max(F_{ij} + F_{ji} - b_{\text{old}}, \sum_x F_{xj} - b_{\text{old}})$ units of bandwidth reserved for backup paths that are simultaneously active with $\wp$. It follows that S_{uv}, the bandwidth available for sharing by $\wp$ on (u, v), is given by:

$$S_{uv} = \max(0, G_{uv} - \max(F_{ij} + F_{ji} - b_{\text{old}}, \sum_x F_{xj} - b_{\text{old}})) \quad (2)$$

For both the above cases, the additional bandwidth reservation required on (u, v), if $\wp$ traverses (u, v), is given by $\max(0, b_{\text{new}} - S_{uv})$. The PLR computes a route for $\wp$ such that the least amount of additional backup bandwidth is reserved. This culminates the suboptimal path computation phase of Kini's framework.

Corrective Signaling Stage: Once the PLR has computed a route for $\wp$, it signals for the path to be setup. Note that in the previous stage the route computed for $\wp$ is suboptimal. Since the PLR makes routing decisions on the basis of aggregate link usage information, for every link (u, v), it assumes that all backup paths in the activation set of a facility protected by $\wp$ traverse (u, v). It is possible that only a subset of these paths traverse (u, v), and therefore, the full extent of backup bandwidth sharing possible on (u, v) is obscured.

The corrective signaling stage can partially compensate for the suboptimal routing of the first stage, during signaling of the path. Although the route computed for $\wp$ remains suboptimal, maximum bandwidth sharing along links in the route computed for $\wp$ is ensured. This is accomplished as follows:

The head-end of each link (u, v) maintains a form of state which we refer to as Link to Facility Incidence Map (LTFIM). The LTFIM for (u, v) contains a list of all facilities that are protected by backup paths that traverse (u, v). For each such facility, the LTFIM for (u, v) maintains b_{facility}, which is the total bandwidth required on (u, v) by backup paths protecting that facility. Once a reservation request arrives at the head end of a link (u, v) along the route computed for $\wp$, the entries corresponding to the facilities protected by $\wp$ are located in the LTFIM for (u, v).[6] Recall that the total bandwidth reserved for backup paths on

[6] In Kini's framework, installing $\wp$ and tearing off $\wp_{\text{old}}$ require LTFIM updates. In practice, explicit tearing off is avoided because RSVP uses soft states.

(u, v) is given by G_{uv}. Since the bandwidth required by $\wp$ is b_{new}, G_{uv} must be at least equal to $b_{\text{new}} + b_{\text{facility}}$ for each facility protected by $\wp$. In case it is not so, we increase G_{uv} so that G_{uv} is equal to $b_{\text{new}} + b_{\text{facility}}$ for that facility. Note that for any link (u, v), additional bandwidth is only reserved when necessary, and therefore, $\wp$ benefits from maximum possible bandwidth sharing on (u, v).

5.2 Facility-Based-Computation (FBC) Framework

The key idea behind the FBC framework is that backup path computation is performed by a node that can make optimal bandwidth sharing decisions for that path. This is accomplished by maintaining a form of local state called Facility to Link Incidence Map (FTLIM). Every node j maintains FTLIM for each link adjacent to it and for itself. Each entry in an FTLIM for a facility corresponds to a link (u, v) and contains b_{uv}, which is the amount of bandwidth reserved on (u, v) by backup paths that belong to the activation set for that facility.

Furthermore, in FBC, each link of the topology maintains, logically disjoint, backup and primary pools [1]. A predetermined[7] percentage of each link is reserved for use by the primary paths and the remaining bandwidth is available for use by the backup paths. We refer to the amount of bandwidth constituting the backup pool on a link (u, v) as B_{uv}. We further define two indicator variables: I_{uv}^{new} which equals 1 if $\wp$ traverses (u, v), and is zero otherwise; and I_{uv}^{old} which equals 1 if $\wp_{\text{old}}$ traverses (u, v), and is zero otherwise. The following provides details of computing a route for $\wp$:

Case 1: $\wp$ Is a Next-Hop Path That Spans a Link (i, j). Recall from section 4 that a next-hop path that spans (i, j) is activated if $\{i, j\}$ fails. The FTLIM corresponding to link $\{i, j\}$ is maintained at both node i and node j. In the FBC framework, node j computes the route for $\wp$. It checks the FTLIM for $\{i, j\}$ at node j, and finds out b_{uv}, which is the bandwidth reserved on (u, v) by backup paths that belong to the activation set of $\{i, j\}$. Observe that it is feasible for $\wp$ to traverse (u, v) if and only if B_{uv} is greater than or equal to $b_{\text{new}} + b_{uv} - I_{uv}^{\text{old}} b_{\text{old}}$. Node j computes a route for $\wp$ using feasible links.

Upon routing $\wp$, some FTLIM entries also need to be updated. When $\wp$ is a next-hop path that spans (i, j), node j locates the FTLIM for the facility $\{i, j\}$, and increments[8] the entry corresponding to (u, v) by $I_{uv}^{\text{new}} b_{\text{new}} - I_{uv}^{\text{old}} b_{\text{old}}$. This update must also be made to the FTLIM for $\{i, j\}$ maintained at node i. This is accomplished during signaling of $\wp$. Since node i is the PLR for $\wp$, node j must communicate the route computed for $\wp$ to node i, so that node i can signal $\wp$. At the same time, node i can update its FTLIM for $\{i, j\}$.

Case 2: $\wp$ Is a Next-Next-Hop path That Spans a Link (i, j) and a Link (j, k). Recall from section 4 that a next-next-hop path that spans (i, j) and (j, k) is activated if either $\{i, j\}$ or node j fails. The FTLIM for node j is maintained at

[7] The primary and backup pools are statically assigned by the network administrator and do not change afterwards.

[8] Since I_{uv}^{new} and I_{uv}^{old} can be 0 or 1 independently, FTLIM update may result in increase or decrease in the value of the entry.

node j and the FTLIM for $\{i, j\}$ is maintained at both node i and node j. In the FBC framework, node j computes the route for $\wp$. The FTLIMs for node j and $\{i, j\}$ contain values of b_{uv} for every link (u, v). Let $b^{\star}_{uv}$ be the maximum of all such values. Therefore, $\wp$ can traverse (u, v) if and only if the backup pool on (u, v) is greater than or equal to $b_{\text{new}} + b^{\star}_{uv} - I^{\text{old}}_{uv} b_{\text{old}}$. Node j uses this information to compute a route for $\wp$ that comprises only feasible links.

As in Case 1, node j increments the entries corresponding to (u, v) by an amount $I^{\text{new}}_{uv} b_{\text{new}} - I^{\text{old}}_{uv} b_{\text{old}}$, in the FTLIMs for the facilities $\{i, j\}$ and node j. Once again, such updates are also made to the FTLIM for $\{i, j\}$ maintained at node i during signaling of $\wp$.

An important feature of FBC is that no explicit bandwidth reservations are made on the links included in the route computed for a backup path. Since the node computing $\wp$ is aware of the bandwidth reservations of backup paths that may be simultaneously active with $\wp$, it ensures that there is no overbooking of bandwidth. Note that this implicitly results in optimal sharing of bandwidth between backup paths that belong to different activation sets.

5.3 NPP: Facility Based Computation Using Aggregate Information

We now present NPP, a new restoration routing framework that draws upon the advantages of Kini's framework and the FBC framework to achieve more efficient bandwidth sharing.

Recall that in Kini's framework, we use aggregate link usage information to compute a route for $\wp$. That is, for this computation, we assume that all backup paths in the same activation set as $\wp$ will traverse a given link (u, v). In actuality, however, only a subset of such paths will be traversing (u, v). Thus, we make a conservative estimate of the bandwidth that can be shared by $\wp$ on (u, v). Therefore, the route computed for $\wp$ is suboptimal. Note that the corrective signaling stage can partially compensate for the suboptimal routing of the initial stage. This is accomplished by reserving the precise amount of bandwidth required by $\wp$ during its signaling. This route is still suboptimal: since the route selection was based on aggregate information, another route that would have minimized the additional bandwidth reservation may have been rejected.

In contrast, the FBC framework computes optimal backup paths using local state. Moreover, perfect bandwidth sharing along the optimally computed paths is also ensured. However, a major disadvantage of the FBC framework is the static allocation of active and backup pools. Even carefully selected pools can remain under-utilized. This under-utilization represents a significant resource wastage and results in a greater number of LSP requests being rejected.

NPP does not suffer from the disadvantages associated with Kini's framework and the FBC framework. Similar to the FBC framework, it shifts the computation of a backup path from the PLR to the node that can make optimal bandwidth sharing decisions for that path. The NPP framework is similar to Kini's framework in that it makes use of aggregate link usage information. The residual capacity of every link in the network represents a single pool of

bandwidth in this framework, as opposed to being divided into logically disjoint primary and backup pools. Furthermore, in the NPP framework nodes are required to maintain the LTFIM and FTLIM as they did in section 5.1 and section 5.2, respectively.

To explain routing a backup path $\wp$, as in the following, we use the same definitions for b_{uv}, I_{uv}^{old}, and I_{uv}^{old} as in the FBC framework:

Case 1: $\wp$ Is a Next-Hop Path That Spans a Link (i, j). Recall from section 4 that a next-hop path that spans (i, j) is activated if $\{i, j\}$ fails. The FTLIM corresponding to link $\{i, j\}$ is maintained at both node i and node j. In NPP, node j computes the route for $\wp$. It checks the FTLIM for $\{i, j\}$ at node j, and finds out b_{uv}, which is the bandwidth reserved on (u, v) by backup paths that belong to the activation set of $\{i, j\}$. Observe that it is feasible for $\wp$ to traverse (u, v) if and only if G_{uv} is greater than or equal to $b_{\text{new}} + b_{uv} - I_{uv}^{\text{old}} b_{\text{old}}$. Node j computes a route for $\wp$ using feasible links.

As in the FBC framework, FTLIM entries need to be updated. Node j increments the entries corresponding to (u, v) by an amount $I_{uv}^{\text{new}} b_{\text{new}} - I_{uv}^{\text{old}} b_{\text{old}}$, in the FTLIM for $\{i, j\}$. As in the FBC case, such updates are also made to the FTLIM for $\{i, j\}$ maintained at node i during signaling of $\wp$.

Case 2: $\wp$ Is a Next-Next-Hop Path That Spans a Link (i, j) and a Link (j, k). Recall from section 4 that a next-next-hop path that spans (i, j) and (j, k) is activated if either $\{i, j\}$ or node j fails. The FTLIM for node j is maintained at node j and the FTLIM for $\{i, j\}$ is maintained at both node i and node j. In the FBC framework, node j computes the route for $\wp$. The FTLIMs for node j and $\{i, j\}$ contain values of b_{uv} for every link (u, v). Let $b_{uv}^{\star}$ be the maximum of all such values. Therefore, $\wp$ can traverse (u, v) if and only if G_{uv} is greater than or equal to $b_{\text{new}} + b_{uv}^{\star} - I_{uv}^{\text{old}} b_{\text{old}}$. Node j uses this information to compute a route for $\wp$ that comprises only feasible links.

Once again, node j increments the entries corresponding to (u, v) by an amount $I_{uv}^{\text{new}} b_{\text{new}} - I_{uv}^{\text{old}} b_{\text{old}}$, in the FTLIMs for the facilities $\{i, j\}$ and node j. Such updates are also made to the FTLIM for $\{i, j\}$ maintained at node i during signaling of $\wp$. Thus, the mechanism to update FTLIMs in NPP framework is quite similar to the mechanism used in FBC framework.

In NPP, the bandwidth reservations on links, along the route computed for $\wp$, are made exactly like those in Kini's framework. That is, for each link (u, v), the head-end node maintains and updates the LTFIM. Once a reservation request arrives at the head end of a link (u, v), along the route computed for $\wp$, the entries corresponding to the facilities protected by $\wp$ are located in the LTFIM for (u, v). The total bandwidth reserved for backup paths on (u, v) is given by G_{uv}. Since the bandwidth required by $\wp$ is b_{new}, G_{uv} must be at least equal to $b_{\text{new}} + b_{\text{facility}}$ for each facility protected by $\wp$. In case it is not so, we increase G_{uv} so that G_{uv} is equal to $b_{\text{new}} + b_{\text{facility}}$ for that facility. Note that for any link (u, v), additional bandwidth is only reserved when necessary, and therefore, $\wp$ benefits from maximum possible bandwidth sharing on (u, v).

6 Simulation Experiments

In this section, we describe the simulation experiments that depict the benefits of NPP over existing frameworks. We conduct a set of experiments and compare the total number of accepted LSP requests and the total bandwidth placed in NPP, FBC and Kini's frameworks. Results for these statistics are presented for element protection. For simulations, a homogeneous network topology is adapted from the network used in [8]. It represents the Delaunay triangulation for the twenty largest metros in continental United States [8]. All the links in the network are symmetric with each link having a capacity of 120 units. Each node in the network may be an LSP ingress or egress node. Therefore, there are 380 possible ingress-egress pairs in the network. LSP requests arrive one by one, where the LSP ingress and egress nodes are chosen randomly from amongst all ingress-egress pairs. The bandwidth demand for an LSP request is uniformly distributed between 1 and 6 units, and the call holding time for each LSP request is infinite. For each LSP request, if it is possible to route the requisite primary and locally restorable facility backup paths, the LSP request is accepted and the associated bandwidth reservations are made on the network; otherwise, the LSP request is rejected. We calculate the number of LSP requests that are successfully placed and the total bandwidth demand associated with these LSPs. We conducted 100 experiments with randomly selected ingress-egress pairs. In each experiment, the primary paths are computed using link costs given by the inverse of residual bandwidth available for primary paths on respective links.

We computed the average of the total number of LSPs and the total bandwidth associated with placed LSPs in these hundered experiments. Fig. 4 shows the number of LSPs placed in NPP in comparison with FBC and Kini's frameworks. It is expected that efficient bandwidth sharing results in better network utilization and hence a greater number of accepted LSP requests. Therefore, NPP – that performs better bandwidth sharing – accepts 775 requests on average compared to 693 requests accepted by FBC framework and 573 requests accepted by Kini's framework. Moreover, the sum of bandwidths for these accepted LSP requests is higher in NPP, as shown in Fig. 4. Experiments on other networks with different sets of LSP requests also indicate similar performance gains when NPP is used.

In our simulations for FBC, a parameter of particular interest is the proportion of bandwidth allocated for primary and backup pools. The value of this parameter is fixed throughout and affects the LSP placement in the network. We used a value for this parameter calculated in the following manner: twenty evenly distributed backup pool percentages between 5 and 100 are used and the one that gives maximum placement of LSPs requests is selected for all the experiments.

7 Conclusions

We investigated the problem of online routing of bandwidth guaranteed LSPs with local restoration. We presented NPP, a new framework for restoration

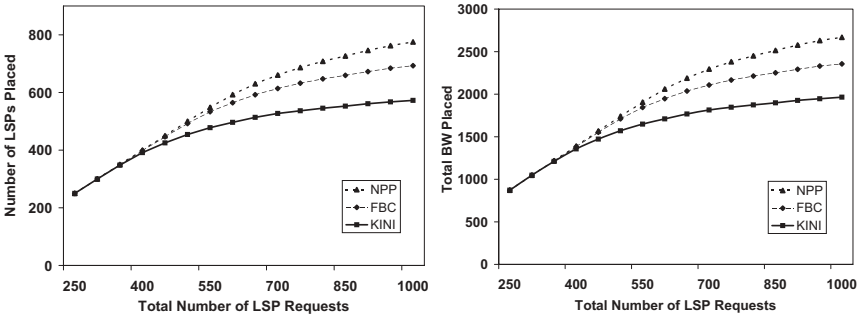


Fig. 4. Total LSPs and total Bandwidth Placed on the Network.

routing using facility backup, and described it in the backdrop of two existing frameworks for restoration routing: Kini and FBC. Kini's framework computes suboptimal routes, since it relies on aggregate rather than complete link usage information to make bandwidth sharing decisions. However, once a path has been computed Kini's framework allowed precise bandwidth reservations to be made during the signaling phase. On the other hand, the FBC framework makes possible optimal path computation by shifting path computation to a node that keeps track of the bandwidth sharing characteristic for that backup path. However, the FBC framework had the disadvantage of statically allocating primary and backup pools, resulting in unutilized bandwidth on certain links. We have shown how NPP draws upon the advantages of both the Kini and FBC frameworks, while avoiding their disadvantages. The key advantage of NPP is that it delivers the bandwidth sharing performance achieved by propagating complete per path link usage information [9], while incurring the significantly reduced routing protocol overhead. Simulation results show that NPP performs significantly better in terms of number of LSPs accepted and total bandwidth placed on the network. For 1000 randomly selected LSP requests on a 20-node homogenous ISP network [8], NPP accepts 775 requests on average compared to 573 requests accepted by Kini's framework of [2] and 693 requests accepted by FBC framework of [1]. Experiments with different sets of LSP requests and on other networks indicate that NPP results in similar performance gains.

References

1. Vasseur, J.-P., Charny, A., Le Faucheur, F., Achirica, J., Leroux, J.-L.: MPLS Traffic Engineering Fast Reroute: Bypass Tunnel Path Computation for Bandwidth Protection. IETF draft. February 2003.
2. Kini, S., Kodialam, M., Sengupta, S., Villamizar, C.: Shared Backup Label Switched Path Restoration. IETF draft. May 2001.
3. Pan, P., Swallow, G., Atlas, A. (Editors): Fast Reroute Extensions to RSVP-TE for LSP Tunnels. IETF draft. August 2004.
4. Vasseur, J.-P., Pickavet, M., Demeester, P.: Network Recovery: Restoration and Protection of Optical, SONET-SDH, IP and MPLS. Morgan Kaufmann. Elsevier. 2004.

5. Awduche, D., Chiu, A., Elwalid, A., Widjaja, I., Xiao, X.: Overview and Principles of Internet Traffic Engineering. RFC 3272. May 2002.
6. Kodialam, M., Lakshman, T. V.: Dynamic Routing of Locally Restorable Bandwidth Guaranteed Tunnels using Aggregated Link Usage Information. Proceedings of IEEE Infocom. pp. 376–385. 2001.
7. Li, L., Buddhikot, M. M., Chekuri, C., Guo, K.: Routing Bandwidth Guaranteed Paths with Local Restoration in Label Switched Networks. Proceedings of IEEE ICNP. pp 110–120. 2002.
8. Norden, S., Buddhikot, M. M., Waldvogel, M. , Suri, S.: Routing Bandwidth Guaranteed Paths with Restoration in Label Switched Networks. Proceedings of IEEE ICNP. pp 71–79. 2001.
9. Kodialam, M., Lakshman, T. V.: Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration. Proceedings of IEEE Infocom. pp 902–911. 2000.
10. Swallow, G.: MPLS Advantages for Traffic Engineering. IEEE Communications Magazine. pp 54–57. vol. 37. no. 12. December 1999.
11. Iannaccone, G., Chuah, C. N., Bhattacharrya, S., Diot, C.: Feasibility of IP Restoration in a Tier-1 Backbone. IEEE Network. pp. 13–19. vol. 18. no. 2. March 2004.
12. Davie, B., Rekhter, Y.: MPLS Technology and Applications. Morgan Kaufmann. San Francisco, CA. 2000.
13. Alcatel: Traffic Engineering Solutions for Core Networks. White Paper. July 2001.
14. Apostolopoulos, G., Guerin, R., Kamat, S., Tripathi, S. K.: Quality of Service Routing: A Performance Perspective. The Proceedings of ACM SIGCOMM. pp 17–28. 1998.
15. Rosen, E., Viswanathan, A., Callon, R.: Multi-Protocol Label Switching (MPLS) Architecture. RFC 3031. January 2001.
16. Charny, A., Vasseur, J.-P.: Distinguish a link from a node failure using RSVP Hellos extensions. IETF draft. October 2002.
17. Katz, D., Kompella, K., Yeung, D.: Traffic Engineering (TE) Extensions to OSPF Version 2. RFC 3630. September 2003.
18. Jamoussi, B. (Editor): Constraint-Based LSP Setup using LDP. RFC 3212. January 2002.