Bandwidth Sharing with Primary Paths for Protection Routing in an MPLS Network

Faisal Aslam, Saqib Raza and Zartash Afzal Uzmi
Department of Computer Science
Lahore University of Management Sciences, Pakistan
Email: {faisal,saqibr,zartash}@lums.edu.pk

Young-Chon Kim
Computer Engineering Department
Chonbuk National University, Korea
Email:yckim@chonbuk.ac.kr

***Abstract*— In label-switched networks such as MPLS, protection routing involves computing and setting up the backup paths at the same time when the primary paths are routed. It has previously been shown that two or more backup paths may share bandwidth along common links if such backup paths will never be activated simultaneously. Such sharing between the backup paths leads to reduced bandwidth reservations and, hence improved performance in terms of number of path requests that can be accommodated on the network [1], [2]. We present a novel idea that backup paths may also share bandwidth with certain primary paths, thereby further reducing the overall bandwidth reservations on the network. This results in even more path requests being accommodated on the network.**

Sharing with primary paths is possible with any protection routing framework. To demonstrate this sharing, we use the NPP protection routing framework as an example [1]. We provide the enhancements to the NPP framework needed to exploit sharing with the primary paths. For the enhanced NPP framework, simulation results on various networks confirm that sharing with primary paths indeed results in better network utilization. This increased performance is achieved with bounded local state information and without requiring any additional routing or signaling overhead.

I. Introduction

Multi-Protocol Label Switching (MPLS) has enabled network service providers establish bandwidth guaranteed explicit paths between customer sites without compromising scalability [3]. Ingress routers of an MPLS network classify packets into forwarding equivalence classes and encapsulate them with labels before subsequently forwarding them along pre-computed label switched paths (LSPs). The ability to place bandwidth guaranteed LSPs using MPLS meets one of the basic QoS requirements for supporting emerging applications such as VoIP. Another QoS requirement is that when a network element along a primary LSP fails, the traffic traversing that LSP is switched onto a preset backup LSP. The backup paths are integral to a protection routing framework and are established in advance–at the same time when a primary path is being established [2], [4], [5]. A backup path is activated only after the failure of a facility[1] it is protecting.

A desirable characteristic of a protection routing framework is minimal latency in switching the traffic onto backup paths when a failure occurs. In *local protection*, the node which diverts traffic from the primary path to the backup path–called the Point of Local Repair (PLR)–is a node that is immediately upstream the failed facility. It is well-known that MPLS *local protection* meets the requirements of real-time applications with recovery times comparable to those of SONET rings [1], [2], [4], [6]. In this paper, we focus on local protection while emphasizing that the idea of sharing with primary paths can also be used with other protection routing frameworks that may not use local protection. A backup path, that emanates from a PLR on the primary path, merges back with the primary path at a downstream node referred to as Merge Point (MP) [7]. In this paper, we consider *many-to-one* protection approach, whereby a PLR maintains a single backup path to protect a set of primary LSPs traversing the triplet (PLR, facility, MP). In many-to-one protection approach, the MP of a backup path is a node that is immediately downstream the facility being protected by that backup path.

Backup provisioning requires bandwidth reservation along the preset backup paths, before the failure occurs, thereby reducing the total number of LSPs that can otherwise be placed on the network. To minimize this reduction, we allow backup paths protecting different facilities to share bandwidth assuming that different network facilities will not fail simultaneously [2], [4], [8].[2] The primary goal in protection routing is to maximize the bandwidth sharing among paths that will not be activated simultaneously. A number of schemes and frameworks have previously been studied for sharing bandwidth along the backup paths [1], [4], [10]. To the best of our knowledge, none of the existing schemes have considered sharing with the primary paths. The key contribution of this paper is to realize that the backup paths protecting a node may share bandwidth with the primary paths originating and terminating from that node.

The rest of the paper is organized as follows: Section II provides the background information and the problem definition. Bandwidth sharing and activation sets are explained in section III. Section IV details the network information used in the original NPP framework and the extended NPP framework, either stored locally at the nodes or propagated

This work was supported in part by a grant from Cisco Systems under the University Research Program, Chonbuk National University, KOSEF through OIRC project, and IITA.

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

[2]This assumption is supported by a recent study on the characterization of network failures in an IP backbone, which revealed that more than 85% of the unplanned network failures affect a single link or a single node [9].

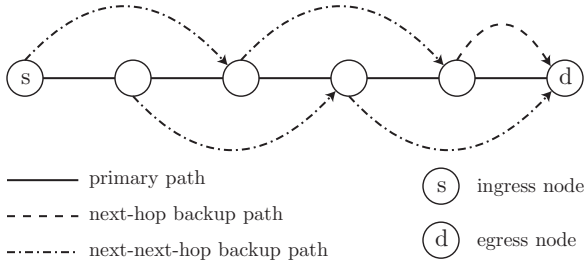


Fig. 1. Local Protection for Single Element Failure

through routing protocols and their extensions. Section V describes the path computation and signaling mechanisms for both the original NPP and the extended NPP framework. The comparative performance achieved by sharing with primary paths is depicted by simulations in section VI which also considers the scalability issues in detail. Lastly, in section VII, we draw the conclusions.

II. Background and Problem Description

In this paper, we use the *single element protection* fault model described in [1] and [11]. Under this model, we setup local backup paths that provide protection against the failure of a single network element, either a node or a link. Towards this end, we recollect the definitions of the two types of backup paths from [1] and [7].

Definition 1: A next-hop path that spans a link (i,j)[3] is a backup path which:

a) originates at node i,
b) merges with the primary LSP(s) at node j, and
c) is activated only when $\{i,j\}$ fails providing protection for one or more primary LSPs that traverse (i,j) and terminate at $node\ j$.

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) is activated when $\{i,j\}$ or $node\ j$ fails providing protection for one or more primary LSPs that traverse the triplet (i,j,k).

Fig. 1 depicts local protection 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 protection 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.

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

In order to serve an LSP request, a bandwidth guaranteed primary path must be setup along with all backup paths that provide local protection using many-to-one approach. If the routing algorithm 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 and the network state is restored to what it was before the arrival of this LSP request. The next LSP request arrives only after the current LSP request has either been accepted or rejected.

For computing the backup paths, our goal is to optimize network utilization such that the network can accept maximum number of LSP requests. A heuristic approach to accommodate maximum number of LSPs on the network is to maximize bandwidth sharing. This is equivalent to minimizing the additional bandwidth reserved for the primary and backup paths for each incoming LSP request, and it is this approach that we use in this paper. We use the following procedure when an LSP request arrives at a node s_k. First, the ingress node s_k computes a primary path for an incoming request; if such a path can not be found, the LSP request is rejected, otherwise s_k signals this primary path. Next, the nodes along the primary path compute corresponding backup paths. If it turns out that the primary path can not be fully protected due to resource constraints, the request is rejected. If, on the other hand, the algorithm is successful in computing all the backup paths, the backup paths are also established within the network. For servicing each LSP request, we try to select routes that minimize the additional bandwidth to be reserved on the network in order to service this request.

III. Bandwidth Sharing and Activation Sets

Our goal of maximizing bandwidth sharing requires an understanding of mechanism by which various paths can share the bandwidth. Each facility is protected by a set of backup paths which are activated when this facility fails. Failure of a facility may also cause deactivation of some primary paths. We define the set of primary and backup paths that will simultaneously be active, when a facility would fail, as the *activation set* of that facility. To formulate the notion of activation sets, we use the following notation:

P : Set of all primary paths.
P_j^{o} : Set of all primary paths originating at $node\ j$.
P_j^{t} : Set of all primary paths terminating at $node\ j$.
μ_{ij} : The next-hop path that spans (i,j);
$nnhop_{ijk}$: The next-next-hop path that spans (i,j) and (j,k).
ω_{ij} : Set of next-next-hop paths that span (i,j) and $(j,x) \forall x \neq i$; $\omega_{ij} = \bigcup_k nnhop_{ijk}$.

Next, we identify the following $m+n+1$ mutually exclusive network states under the single element failure model:

a) *Default state*, when there is no failure in the network,

[3]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).

b) *Link Failure state*, when a single link has failed, and
c) *Node Failure state*, when a single node has failed.

While the network states are mutually exclusive, the corresponding activation sets, given below, may not be disjoint:
Default Activation Set: This set consists of paths that are active when the network is in default state. This is a no-failure state and thus, the default activation set is equal to P.
Activation Set for $\{i,j\}$*:* Under the element protection fault model, a link is protected by next-hop and next-next-hop paths. Therefore, when $\{i,j\}$ fails, all next-hop and next-next-hop paths that protect $\{i,j\}$ are activated [1]. Thus, the activation set for a link $\{i,j\}$ is:

$$S_{ij} = \mu_{ij} \cup \mu_{ji} \cup \omega_{ij} \cup \omega_{ji} \cup P$$

Activation Set for node j*:* Only next-next-hop paths protect against failure of a node. When $node\ j$ fails, all such next-next-hop paths are activated. Thus S_j, the activation set for $node\ j$, is given by:

$$S_j = (\bigcup_i \omega_{ij}) \cup (P - P_j^{\mathrm{t}} - P_j^{\mathrm{o}})$$

Two paths will not be simultaneously active unless they are in the same activation set. It follows that such paths can share bandwidth with each other along common links. In contrast, paths that are simultaneously active must make bandwidth reservations that are exclusive of each other. Thus, we deduce:

a) The default activation set constrains the primary paths not to share bandwidth with each other.
b) $P \subseteq S_{ij}$ implies that none of the backup paths in S_{ij} can share bandwidth with any primary path.
c) When $node\ j$ fails, all primary LSPs that originate or terminate at $node\ j$ cease to exist. Such primary paths are not included in S_j and may, therefore, share bandwidth with backup paths in S_j.

This sharing with primary paths not included in S_j is in addition to the usual sharing among the backup paths [1].

IV. Network Information

Maximum bandwidth sharing can be achieved if the node computing a backup path has complete information about the local and propagated network state [12]. However, making such information available to all nodes incurs significant protocol overhead in terms of network utilization, memory, update-processing and associated context switching [13]. Network information is of two types: stored locally at the nodes or advertised through routing protocols and their extensions.

A. Propagated Network Information

The NPP framework relied on propagating the following aggregate per link network usage information for each link (i,j) in the network, using traffic engineering extensions to existing link state routing protocols [14]:

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

In the extended NPP framework, we again rely on propagating aggregate link usage information and, therefore, there is no additional routing overhead associated with the extended NPP framework.

B. Local State Information

To achieve perfect sharing among backup paths, the NPP framework maintains two maps locally stored at network nodes which are best described by using the following notation:

α_{ij} : Set of all primary paths that traverse (i,j)
β_{ij} : Set of all backup paths that traverse (i,j)
$\| x \|$: Sum of bandwidths of all paths in a set x

The maps used in the NPP framework are:
1) Backup Facility to Link Incidence Map (BFTLIM): For each facility f, $\mathrm{BFTLIM}^f(u,v)$ is the backup bandwidth that gets activated on (u,v), when facility f fails. That is,

$$\mathrm{BFTLIM}^{\{i,j\}}(u,v) = \| S_{ij} \cap \beta_{uv} \|$$
$$\mathrm{BFTLIM}^{node\ j}(u,v) = \| S_j \cap \beta_{uv} \|$$

2) Backup Link to Facility Incidence Map (BLTFIM): For each link (u,v), $\mathrm{BLTFIM}^{(u,v)}(f)$ is the backup bandwidth that gets activated on (u,v), when facility f fails. Therefore,

$$\mathrm{BLTFIM}^{(u,v)}(\{i,j\}) = \| S_{ij} \cap \beta_{uv} \|$$
$$\mathrm{BLTFIM}^{(u,v)}(node\ j) = \| S_j \cap \beta_{uv} \|$$

Besides using BFTLIMs and BLTFIMs as in the NPP framework [1], the extended NPP framework maintains two additional maps to allow sharing with primary paths; these two maps are described below:
3) Primary Facility to Link Incidence Map (PFTLIM): For each facility f, $\mathrm{PFTLIM}^f(u,v)$ is the total bandwidth of primary paths with which backup paths protecting f may share bandwidth. This includes all primary paths that either originate or terminate at the facility f. Therefore,

$$\mathrm{PFTLIM}^{\{i,j\}}(u,v) = \| \alpha_{uv} - S_{ij} \|$$
$$\mathrm{PFTLIM}^{node\ j}(u,v) = \| \alpha_{uv} - S_j \|$$

Since $\alpha_{uv} \subseteq P$ and $P \subseteq S_{ij}$, therefore, $\alpha_{uv} - S_{ij}$ is an empty set and hence $\mathrm{PFTLIM}^{\{i,j\}}(u,v) = 0\ \forall \{i,j\},(u,v)$.
4) Primary Link to Facility Incidence Map (PLTFIM): For each link (u,v), $\mathrm{PLTFIM}^{(u,v)}(f)$ is the total bandwidth of primary paths traversing (u,v), which can be shared by backup paths in the activation set of f.

$$\mathrm{PLTFIM}^{(u,v)}(\{i,j\}) = \| \alpha_{uv} - S_{ij} \|$$
$$\mathrm{PLTFIM}^{(u,v)}(node\ j) = \| \alpha_{uv} - S_j \|$$

C. Local State Storage

A given node stores only a small number of the maps given above. Specifically, a $node\ j$ stores $\mathrm{BFTLIM}^{node\ j}(\cdot)$, and $\mathrm{BFTLIM}^{\{j,x\}}(\cdot)\forall x$. Thus, $node\ j$ will store $(d^j + 1)$ BFTLIMs, where d^j is the degree of $node\ j$. Furthermore, a $node\ j$ stores d^j BLTFIMs corresponding to each outgoing link, i.e., $\mathrm{BLTFIM}^{(j,x)}(\cdot)\forall x$.

Each $node\ j$ in the network maintains a single primary facility to link incidence map given by $\mathrm{PFTLIM}^{node\ j}(u,v)$, because $\mathrm{PFTLIM}^{\{i,j\}}(u,v) = 0\ \forall\{i,j\},(u,v)$. Each $node\ j$ also maintains d^j PLTFIMs corresponding to each outgoing link, i.e., $\mathrm{PLTFIM}^{(j,x)}(\cdot)\forall x$.

V. Path Computation and Signaling

When an LSP request arrives at an ingress node, computation of primary and backup paths is needed before accepting or rejecting the request. The primary path is computed by the ingress node and the backup paths are computed by intermediate nodes during the signaling of the primary path.

A. Backup Path Computation

For optimal sharing, the NPP and the extended NPP frameworks compute $nnhop_{ijk}\ \forall i,k$ and $\mu_{ij}\ \forall i$ at $node\ j$ which has maximal information of the activation set of the facilities protected by these backup paths.

Suppose we need to compute a backup path $\wp_{\mathrm{new}}$ with bandwidth demand b_{new} originating from $node\ i$ which is a PLR on the primary path. Assume that a backup path $\wp_{\mathrm{old}}$, with cumulative bandwidth demand b_{old},[4] was already protecting the same (PLR, facility, MP) triplet. Then we have, $b_{\mathrm{new}} = b_k + b_{\mathrm{old}}$. To elucidate the computation of $\wp_{\mathrm{new}}$, we use the following two indicator variables:

$$I_{uv}^{\mathrm{new}} = \begin{cases} 1 & \text{if } \wp_{\mathrm{new}} \text{ traverses } (u,v) \\ 0 & \text{otherwise} \end{cases}$$

$$I_{uv}^{\mathrm{old}} = \begin{cases} 1 & \text{if } \wp_{\mathrm{old}} \text{ traverses } (u,v) \\ 0 & \text{otherwise} \end{cases}$$

The path $\wp_{\mathrm{new}}$ may be a next-hop path or a next-next-hop path, leading to the following two cases:

Case 1: When $\wp_{\mathrm{new}}$ is a next-hop path that spans a link (i,j), it must not share bandwidth with the paths in the activation set of $\{i,j\}$. The computation of $\wp_{\mathrm{new}}$ is done at $node\ j$ which can make an optimal sharing decision since it locally maintains $\mathrm{BFTLIM}^{\{i,j\}}(u,v)$ for each link (u,v). Actual computation is a 3-step process. Firstly, $node\ j$ determines shareable bandwidth on each link (u,v) as:

$$\zeta_{\{i,j\}}^{(u,v)} = G_{uv} - \mathrm{BFTLIM}^{\{i,j\}}(u,v)$$

Secondly, the additional bandwidth needed on (u,v), if next-hop path $\wp_{\mathrm{new}}$ were to traverse (u,v) is determined:

$$\varrho_{\{i,j\}}^{(u,v)} = \max\left(0, b_{\mathrm{new}} - I_{uv}^{\mathrm{old}} b_{\mathrm{old}} - \zeta_{\{i,j\}}^{(u,v)}\right)$$

Finally, $node\ j$ uses the following cost function[5] to compute the next-hop path $\wp_{\mathrm{new}}$:

$$\theta_{(i,j)}^{(u,v)} = \begin{cases} \varrho_{\{i,j\}}^{(u,v)} & R_{uv} \geq \varrho_{\{i,j\}}^{(u,v)} \\ \infty & \text{otherwise} \end{cases}$$

[4]If a backup path protecting the same (PLR, facility, MP) triplet did not exist, then $b_{\mathrm{old}} = 0$.

[5]The cost function assigns a cost to each link in the network. The individual costs are then used to find the total cost incurred in routing a backup LSP between the PLR and the MP.

After computing a least cost backup path, $node\ j$ increments $\mathrm{BFTLIM}^{\{i,j\}}(u,v)$, for every link (u,v), by an amount equal to $I_{uv}^{\mathrm{new}} b_{\mathrm{new}} - I_{uv}^{\mathrm{old}} b_{\mathrm{old}}$. Furthermore, $\mathrm{BFTLIM}^{\{i,j\}}(u,v)$ stored at $node\ i$ is incremented by the same amount, during the signaling of next-hop backup path $\wp_{\mathrm{new}}$. This procedure is the same as was used in the NPP framework [1].

Case 2: When $\wp_{\mathrm{new}}$ is a next-next-hop path that spans a link (i,j) and a link (j,k), it must not share bandwidth with the paths in the activation sets of $\{i,j\}$ and of $node\ j$. Maximal information about these activation sets is locally stored at $node\ j$ which computes the next-next-hop backup path $\wp_{\mathrm{new}}$. Since primary incidence maps are used for path computation in this case, the extended NPP framework uses a different procedure compared to that used in NPP framework, for the computation of next-next-hop path $\wp_{\mathrm{new}}$.

For backup path computation, $node\ j$ determines shareable bandwidth on each link (u,v) and the additional bandwidth needed on (u,v), if $\wp_{\mathrm{new}}$ were to traverse (u,v). The shareable and additional bandwidths are computed for two different activation sets, one for each facility being protected by $\wp_{\mathrm{new}}$.

If (i,j) were to fail, the shareable bandwidth available and the additional bandwidth to be reserved, respectively, are:

$$\zeta_{\{i,j\}}^{(u,v)} = G_{uv} - \mathrm{BFTLIM}^{\{i,j\}}(u,v)$$
$$\varrho_{\{i,j\}}^{(u,v)} = \max\left(0, b_{\mathrm{new}} - I_{uv}^{\mathrm{old}} b_{\mathrm{old}} - \zeta_{\{i,j\}}^{(u,v)}\right)$$

If, however, $node\ j$ were to fail, we compute the shareable bandwidth as:

$$\zeta_{node\ j}^{(u,v)} = G_{uv} - \mathrm{BFTLIM}^{node\ j}(u,v) + \mathrm{PFTLIM}^{node\ j}(u,v)$$

which is where the sharing with primary paths is realized by the extended NPP framework. The additional bandwidth to be reserved on (u,v) when protecting $node\ j$ is given by:

$$\varrho_{node\ j}^{(u,v)} = \max\left(0, b_{\mathrm{new}} - I_{uv}^{\mathrm{old}} b_{\mathrm{old}} - \zeta_{node\ j}^{(u,v)}\right)$$

Finally, the cost function used by $node\ j$ is:

$$\theta_{node\ j}^{(u,v)} = \begin{cases} \max\left(\varrho_{\{i,j\}}^{(u,v)}, \varrho_{node\ j}^{(u,v)}\right) & \\ & R_{uv} \geq \max\left(\varrho_{\{i,j\}}^{(u,v)}, \varrho_{node\ j}^{(u,v)}\right) \\ \infty & \text{otherwise} \end{cases}$$

After computing a least cost backup path, $node\ j$ increments $\mathrm{BFTLIM}^{\{i,j\}}(u,v)$ and $\mathrm{BFTLIM}^{node\ j}(u,v)$, for every link (u,v), by an amount equal to $I_{uv}^{\mathrm{new}} b_{\mathrm{new}} - I_{uv}^{\mathrm{old}} b_{\mathrm{old}}$. Similarly, $\mathrm{BFTLIM}^{\{i,j\}}(u,v)$ stored at $node\ i$ is also incremented when the next-next-hop backup path $\wp_{\mathrm{new}}$ is being signaled. During the computation and signaling of backup paths, no changes are made to the PFTLIMs or PLTFIMs anywhere in the network. Such changes to primary incidence maps (PFTLIM and PLTFIM) are made only during the signaling of primary paths, and will be explained in section V-C.

B. Backup Path Signaling

A node computing the backup path signals such path to the PLR for installation on the network, using the procedure similar to that given in [8]. Once again, we consider the following two cases:

Case 1: $\wp_{\text{new}}$ is a next-hop path that spans a link (i, j). After path computation and local state update, $node\ j$ signals the computed path to $node\ i$ which is the PLR for $\wp_{\text{new}}$. The signal from $node\ j$ to $node\ i$ includes the facility $\{i, j\}$, the requisite bandwidth b_{new}, and the explicit route object (ERO)[6] [8]. The responsibility of installation of the backup path lies with $node\ i$ which, after updating its own local state, requests reservation of bandwidth along all the links traversed by $\wp_{\text{new}}$.

When a $node\ u$ receives a reservation request along the link (u, v), it increments locally stored $\text{BLTFIM}^{(u,v)}(\{i, j\})$ by $b_{\text{new}} - I_{uv}^{\text{old}} b_{\text{old}}$. Afterwards, $node\ u$ updates G_{uv} for every facility f as,

$$G_{uv} = \max_f \text{BLTFIM}^{(u,v)}(f)$$

That is, G_{uv} is updated to represent the the maximum value in the map $\text{BLTFIM}^{(u,v)}(\cdot)$. Note that G_{uv} may or may not change from its previous value. That is, for any link (u, v), additional bandwidth will only be reserved when necessary, which leads to maximum sharing.

Case 2: $\wp_{\text{new}}$ is a next-next-hop path that spans a link (i, j) and a link (j, k). After path computation and local state update, $node\ j$ signals the computed path to $node\ i$ which is the PLR for $\wp_{\text{new}}$. The signal from $node\ j$ to $node\ i$ includes the facilities $\{i, j\}$ and $node\ j$ being protected by $\wp_{\text{new}}$, the requisite bandwidth b_{new}, and the ERO. Recall from section V-A, the PLR $node\ i$ updates $\text{BFTLIM}^{\{i,j\}}(u, v)$ after receiving signal from $node\ j$. Afterwards, $node\ i$ requests reservation of bandwidth along all the links traversed by $\wp_{\text{new}}$.

When a $node\ u$ receives a reservation request along the link (u, v), it increments the entries $\text{BLTFIM}^{(u,v)}(\{i, j\})$ and $\text{BLTFIM}^{(u,v)}(node\ j)$ by $b_{\text{new}} - I_{uv}^{\text{old}} b_{\text{old}}$. Afterwards, $node\ u$ updates G_{uv} for every facility f as:

$$G_{uv} = \max_f \left(\text{BLTFIM}^{(u,v)}(f) - \text{PLTFIM}^{(u,v)}(f)\right)$$

where $\text{PLTFIM}^{(u,v)}(f)$ is zero when the facility f represents a link. Once again, G_{uv} may or may not change from its previous value.

C. Primary Path Signaling

The primary path is computed and signaled by the ingress node. During the signaling, part of the local network state is updated at each node along the primary path. Consider an LSP request with ingress node s_k, the LSP egress node d_k, and a bandwidth demand b_k. While placing this primary path, the following entries are incremented by an amount b_k:

a) $\text{PFTLIM}^{s_k}(u, v)$ at the ingress $s_k, \forall(u, v)$ along the primary path.

[6]The ERO enlists all the links along the computed backup path $\wp_{\text{new}}$.

b) $\text{PLTFIM}^{(u,v)}(s_k)$ at each intermediate $node\ u$ that is requested to route the primary along (u, v).

c) $\text{PLTFIM}^{(u,v)}(d_k)$ at each intermediate $node\ u$ that is requested to route the primary along (u, v).

d) $\text{PFTLIM}^{d_k}(u, v)$ at the egress $d_k, \forall(u, v)$ along the primary path.

It is important to note that above updates are carried out locally without requiring any extraneous signaling.

D. Implications of Requests Reordering

We noticed that backup paths protecting a node may share bandwidth with primary paths that either originate or terminate at that node. However, an existing backup path protecting $node\ j$ can not benefit from an increased sharing opportunity that may result from a later placement of primary paths originating or terminating at $node\ j$. This results in backup bandwidth sharing with primary paths being sensitive to the order in which LSP requests arrive. To make such sharing, and hence the bandwidth reservations, independent of LSP request order, each intermediate $node\ u$ along the primary path traversing (u, v) updates G_{uv} during primary path signaling after the PLTFIMs are updated. That is, we set:

$$G_{uv} = \max_f \left(\text{BLTFIM}^{(u,v)}(f) - \text{PLTFIM}^{(u,v)}(f)\right)$$

This optimization may lead to a decrease in G_{uv} leaving additional remaining bandwidth R_{uv} for accommodating future LSP requests.

VI. Simulations and Performance

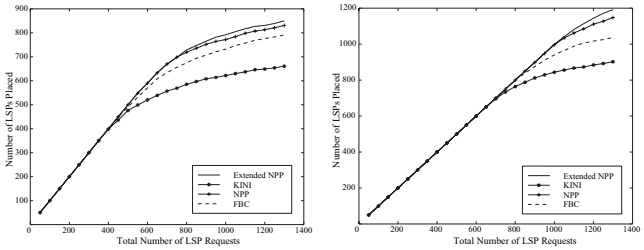
We report performance in terms of number of LSPs placed, for two different networks. Network 1 is a 15 node heterogeneous topology adapted from the network in [2]. The links in the core have a capacity of 480 units in either direction, while other links have a capacity of 120 units in each direction. Network 2 is a homogeneous topology adapted from [5], and represents the Delaunay triangulation for the twenty largest metros in continental US [5]. Each unidirectional link in the network has a capacity of 120 units.

A. Simulation Experiments

The traffic matrix consists of a series of LSP requests along with their requisite bandwidth. For each LSP request, ingress and egress nodes are chosen randomly and the bandwidth demand is uniformly distributed between 1 and 4 units, with an infinite call holding time. Total number of accepted LSPs for the two example networks are reported for NPP, enhanced NPP, Kini, and FBC frameworks [1], in Fig. 2. All the results in figures represent an average taken from 100 experiments for each of which a new traffic matrix is generated.

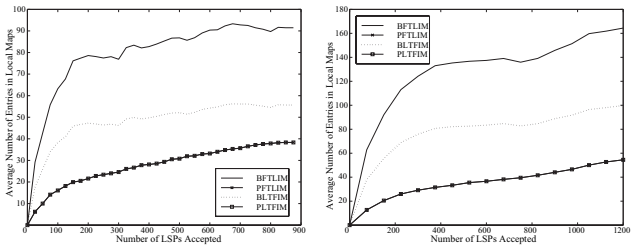
B. Scalability

From the local state storage description in section IV-C, we note that the number of entries in each map is bounded. Specifically, the number of entries in facility to link incidence maps is bounded by the total number of links in the network. Similarly, the number of entries in link to facility incidence



(a) Network 1 (b) Network 2

Fig. 2. Number of accepted LSPs: Network 1 (left), Network 2(right)



(a) Network 1 (b) Network 2

Fig. 3. Average number of entries in four types of locally stored maps. The average is taken over all nodes in the netowrk.

maps is bounded by the total number of facilities–the sum of number of nodes and number of bidirectional links–in the network. Furthermore, the local information is never propagated, and there is no routing protocol overhead in using the extended NPP framework.

During the simulations, we also monitored the actual amount of local state information stored at each node. We assumed that local state maps only store non-zero entries, and a non-existent entry defaults to zero. Each node in the network maintains up to four types of incidence maps: BFTLIM, BLTFIM, PFTLIM, and PLTFIM. The average number of entries in all maps of each type are shown in Fig. 3 against the number of LSPs accepted for placement on the network.

It is worth noting that as the network utilization increases, fewer new entries are added to the incidence maps. The main reason is that the locally stored maps only keep the aggregated information and a new LSP in a well-utilized network is expected to take a path whose entries have previously been entered into the maps. Hence, in a well-utilized network, a new LSP is likely to change the stored entries but it is not likely to create new entries.

VII. Concluding Remarks

We investigated the problem of online routing of primary and backup paths in a label switched network using many-to-one local protection, and demonstrated that backup paths protecting a node are allowed to share bandwidth with primary paths that either originate or terminate at that node. Since no paths originate or terminate at links, it is not possible for those backup paths that only protect a link to share bandwidth with primary paths.

We used the NPP framework as an example and extended it to allow sharing with the primary paths. For the extended NPP framework, we explained the path computation algorithm, signaling mechanism, and the local and propagated network state information. We noticed that sharing with the primary paths allows more sharing in the network, which heuristically results in placement of more traffic. The heuristic is verified by simulations for various network topologies, and improvements, in terms of number of accepted requests, are reported. We observe that the original NPP framework already uses maximal sharing among the backup paths leaving little room for improvement if sharing with primary paths is also allowed.

We also noticed that the extended NPP framework does not require any additional routing overhead. Furthermore, local state information overhead required by the extended NPP framework scales well with the network load: as the network load increases, growth of local state information diminishes, where the amount of local state information is bounded by a value completely determined by the number of links and nodes in the network.

References

[1] F. Aslam, S. Raza, F. R. Dogar, I. U. Ahmad, and Z. A. Uzmi, "NPP: A Facility Based Computation Framework for Restoration Routing Using Aggregate Link Usage Information," in *Proc. QoS-IP*, February 2005, pp. 150–163.

[2] M. S. Kodialam and T. V. Lakshman, "Dynamic Routing of Locally Restorable Bandwidth Guaranteed Tunnels Using Aggregated Link Usage Information," in *Proceedings of Infocom*, April 2001, pp. 376–385.

[3] Alcatel, "White Paper: Traffic Engineering Solutions for Core Networks," July 2001.

[4] L. Li, M. M. Buddhikot, C. Chekuri, and K. Guo, "Routing Bandwidth Guaranteed Paths with Local Restoration in Label Switched Networks," *IEEE JSAC*, vol. 23, no. 2, pp. 437–449, 2005.

[5] S. Norden, M. M. Buddhikot, M. Waldvogel, and S. Suri, "Routing Bandwidth Guaranteed Paths with Restoration in Label Switched Networks," in *Proceedings of ICNP*, November 2001, pp. 71–79.

[6] G. Swallow, "MPLS Advantages for Traffic Engineering," *IEEE Communications Magazine*, vol. 37, no. 12, pp. 54–57, December 1999.

[7] P. Pan and G. Swallow and A. Atlas (Editors), "RFC 4090: Fast Reroute Extensions to RSVP-TE for LSP Tunnels," May 2005.

[8] J.-P. Vasseur, A. Charny, F. L. Faucheur, J. Achirica, and J.-L. Leroux, "Internet Draft: MPLS Traffic Engineering Fast Reroute: Bypass Tunnel Path Computation for Bandwidth Protection," February 2003.

[9] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, "Characterization of Failures in an IP Backbone," in *Proceedings of Infocom*, March 2004, pp. 2307–2317.

[10] M. Kodialam and T. V. Lakshman, "Dynamic Routing of Restorable Bandwidth-Guaranteed Tunnels using Aggregated Network Resource Usage Information," *IEEE/ACM Trans. Networking*, vol. 11, no. 3, pp. 399–410, 2003.

[11] S. Raza, F. Aslam, and Z. A. Uzmi, "Online Routing of Bandwidth Guaranteed Paths with Local Restoration using Optimized Aggregate Usage Information," in *Proceedings of the 2005 IEEE International Conference on Communications (ICC)*, Seoul, Korea, May 2005, pp. 201–207.

[12] M. S. Kodialam and T. V. . Lakshman, "Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration," in *Proceedings of Infocom*, March 2000, pp. 902–911.

[13] G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, "Quality of Service Routing: A Performance Perspective," in *Proc. ACM SIGCOMM*, 1998, pp. 17–28.

[14] D. Katz, K. Kompella, and D. Yeung, "RFC 3630: Traffic Engineering (TE) Extensions to OSPF Version 2," September 2003.