Targeted Distribution of Resource Allocation for Backup LSP Computation

Mohand Yazid SAIDI, Bernard COUSIN
Université de Rennes I - Campus de Beaulieu
IRISA/INRIA, 35042 Rennes Cedex, France
{msaidi,bcousin}@irisa.fr

Jean-Louis LE ROUX
France Télécom
2, Avenue Pierre Marzin, 22300 Lannion, France
jeanlouis.leroux@orange-ftgroup.com

Abstract

Under the hypothesis of single failures in the network, some backup paths cannot be active at the same time because they protect against the failure of different components. Hence, share the bandwidth between such backup paths is central to optimize the bandwidth allocated in the network and to decrease the bandwidth wasting.

In this paper, we propose a novel algorithm, based on Targeted Distribution of Resource[1] *Allocation (TDRA), to compute the backup Label Switched Paths (LSPs) in a distributed MultiProtocol Label Switching (MPLS) environment. Our algorithm is scalable, efficient and capable to protect against the three types of failure risk: node, link and Shared Risk Link Group (SRLG). Indeed, the TDRA algorithm decreases the quantity of information (resource or bandwidth allocation) transmitted in the network with the selection of nodes to be advertised (with the selection of recipient nodes). Furthermore, bandwidth availability is increased by sharing bandwidth between backup LSPs as long as possible.*

Simulations show that the ratio of rejected backup LSPs obtained with the transmission of a small quantity of information in the network is low.

Keywords– *network, local protection, backup LSP, SRLG, MPLS, bandwidth sharing, path computation.*

1. Introduction

Today's applications are very sensitive to the disruption of communications and require more and more bandwidth to operate. Two functionalities of traffic engineering cope with these evolutions: protection and resource optimization.

The protection is the technique which deals with failures to prevent or to decrease the interruption time of communications [10, 12]. It is based on the computation of backup paths which will be used to forward traffic of the affected primary paths upon a failure.

Like the primary paths, the backup paths must reserve resources (especially bandwidth) in order to guarantee their availability after a failure. However (contrarily to the primary paths), the backup paths do not use the reserved resources as long as the protected component (eg. link or node) is not failing. As a result, under the single failure hypothesis, one resource can be shared and allocated for all the backup paths which protect against distinct failure risks (i.e. components which cannot fail at the same time). Indeed, these backup paths cannot claim simultaneously the shared resource since at any time, at most one backup path can be active. Hence, to maximize the resource availability in the network, the backup path computation must account for resource sharing.

With the advent of MultiProtocol Label Switching (MPLS) [13], the two preceding functionalities of traffic engineering are provided effectively.

Firstly, the recovery delay is decreased with the precomputation and pre-configuration of local backup Label Switched Paths (LSPs). Two types of backup LSPs are defined for MPLS local protection [11]: Next HOP backup LSP (NHOP LSP) and Next Next HOP backup LSP (NNHOP LSP). A NHOP LSP (resp. NNHOP LSP) is a backup LSP protecting against link failure (resp. node failure); it is setup between a Label Switched Router (LSR) called Point of Local Repair (PLR) and one LSR called Merge Point (MP) located between the next-hop (resp. next-next-hop) of the PLR and the destination. Such backup LSP bypasses the link downstream (resp. the node downstream) to the PLR on the primary LSP. When a link failure (resp. node failure) is detected by a node, this later activates locally all its NHOP and NNHOP (resp. NNHOP) backup LSPs by switching traffic from the affected primary LSPs to their backup LSPs.

Secondly, MPLS offers large flexibility for path choosing. That allows resource optimization by the selection of paths which maximize the bandwidth sharing, for instance.

In this article, we propose a novel algorithm based on

[1] *Resource* refers to *bandwidth* in this paper.

Targeted Distribution of Resource Allocation (TDRA algorithm) to compute on-line the backup LSPs in a distributed environment. The on-line mode signifies that each computation request is treated as soon as it comes, with no a priori knowledge of future request arrivals. The choice of the distributed environment is motivated by the concern of offering the scalability and reactivity. Thus, with TDRA algorithm, each network node supports one Backup Path Computation Element (BPCE). Each element is then responsible of the computation of backup LSPs protecting against the failure of the following components: the node supporting the BPCE and all its (incoming) adjacent links. With this architecture, certain quantity of resource information should be shared between BPCEs to allow the protection against the failure of the three types of risks: node, link and Shared Risk Link Group (SRLG). With our TDRA algorithm, such information is decreased and sent only to nodes requiring it for the backup LSP computation.

The rest of this paper is organized as follows. Section 2 describes the three types of failure risks. Each failure risk gathers network components which can fail simultaneously into one entity. By relying on the hypothesis that there is at most one failure occurrence at any time, we give the formulas allowing the computation of the minimal protection bandwidth to be reserved on each (unidirectional) link. In section 3, we review works related to the bandwidth sharing. Then, we propose in section 4 our TDRA algorithm which efficiently balances the backup path computations on nodes. Moreover, the size of control information transmitted in the network is low and homogeneously distributed on all the links of the network topology. In the next section, we propose slight modifications to the signaling and routing protocol in order to deploy the TDRA algorithm. In section 6, we analyze the performances of the TDRA algorithm. Finally, section 7 is dedicated to the conclusions.

2. Failure risks

When a physical component failure occurs, several network components may be affected simultaneously. Such network components should be grouped in one entity, called *failure risk*, in order to better manage failures and to optimize the bandwidth allocated in the network. In fact, the network components failing simultaneously determine the set of backup LSPs which can be active at the same time and thus, they determine the minimal bandwidth to be reserved in links for the protection.

Under the hypothesis of single physical failures adopted in this paper, we distinguish three types of logical failure risks: link risk, node risk and SRLG [6]. The first risk corresponds to the risk of a logical link failure due to the breakdown of an exclusive physical component to the logical link (eg. An optical fiber connecting two MPLS routers).

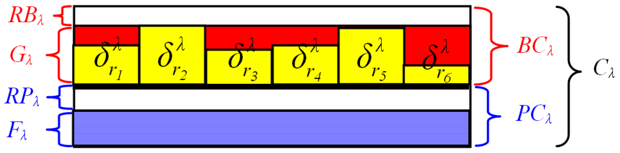


Figure 1. Bandwidth allocation on an arc λ

The second failure risk corresponds to the risk of a logical node failure. Finally, the SRLG corresponds to the risk of simultaneous failures of some logical links[2]. This last risk is defined to address the failure of a physical component (like some optical crossconnects or Data Link components) shared between several logical links.

To determine the minimal protection bandwidth to be reserved on arcs (unidirectional links), [9] defines two concepts: *the Protection Failure Risk Group (PFRG)* and *the protection cost*. The *PFRG* of a given arc λ, noted $PFRG(\lambda)$, is a set composed of all the risks protected by the backup LSPs traversing the arc λ. The protection cost of a risk *r* on an arc λ, noted δ^λ_r, is defined as the cumulative bandwidth of the backup LSPs which will be activated on the arc λ upon a failure of the risk r. For a SRLG risk *srlg* composed of links $(l_1, l_2, .., l_n)$, the protection cost on an arc λ is determined as follow:

$$\delta^\lambda_{srlg} = \sum_{0<i\leq n} \delta^\lambda_{l_i} \qquad (1)$$

To cope with any failure, a minimal quantity of protection bandwidth G_λ must be reserved on the arc λ. Such quantity G_λ is determined as the maximum of the protection costs on the arc λ.

$$G_\lambda = Max_{r\in PFRG(\lambda)}\delta^\lambda_r \qquad (2)$$

In order to better control (explicitly specify) the quantity of bandwidth used for protection and to separate the computation task of primary LSPs from that of backup LSPs, the bandwidth capacity C_λ on each arc λ can be divided in two pools: primary bandwidth pool and protection bandwidth pool (figure 1). The primary bandwidth pool on an arc λ has a capacity PC_λ and it is used to allocate bandwidth for primary LSPs. The protection bandwidth pool on an arc λ has a capacity BC_λ and it is used to allocate bandwidth for backup LSPs.

To ensure the respect of bandwidth constraints, the reserved protection bandwidth on each arc λ must verify:

$$G_\lambda \leq BC_\lambda \qquad (3)$$

[2] The notion of SRLG risk can also be used to cope with double (or more) simultaneous link failures.

To keep inequality (3) valid after the setup of a backup LSP b of bandwidth $bw\ (b)$ which protects against the risks of a given set $FR\ (b)^3$, only the arcs (λ) verifying the following inequality can be selected to be in the LSP b:

$$Max_{r \in FR(b)} \delta_r^\lambda \leq BC_\lambda - bw\ (b) \tag{4}$$

Finally, we define the residual protection bandwidth RB_λ as the quantity of protection bandwidth which is not used on the arc λ. It is determined as follow:

$$RB_\lambda = BC_\lambda - G_\lambda \tag{5}$$

In a similar way, we determine the residual primary bandwidth RP_λ as the difference between the primary capacity PC_λ and the cumulated primary bandwidth F_λ allocated on the arc λ:

$$RP_\lambda = PC_\lambda - F_\lambda \tag{6}$$

3. Related Works

In the last years, various papers proposed algorithms to compute on-line backup LSPs sharing bandwidth. In spite of the SRLG existence in actual networks (especially in optical networks), the majority of the proposed algorithms does not deal with SRLG risks. In this section, we concentrate only on the computation techniques allowing the determination of bandwidth-guaranteed backup LSPs treating all the types of failure risk (link, node and SRLG). Dependently on the used environment, these techniques can be classed in: centralized and distributed techniques.

In a centralized environment, the server can store the network topology and the LSPs structures and properties. With such data, the problem of bandwidth sharing can be formulated using the integer linear programming. An example of such formulation for end-to-end protection is described in [3]. In this formulation, the primary path and its backup path are computed so that the additional bandwidth they need is minimal. Even though this technique increases the bandwidth availability, its utilization is limited to small networks. Indeed, the use of a centralized server does not scale and presents some well known disadvantages like the formation of bottlenecks around the server (non scalable) and the sensitivity to the failure or overload of the server. In addition, the centralized server reactivity is not stable and it decreases significantly after traffic matrix changes.

To get around the previous drawbacks, [2] suggests to flood within IGP-TE protocols [5, 4] the network topology, the primary bandwidth, the capacities and all the protection costs ($\{\delta_r^\lambda\}_{\lambda,r}$) in the network. In this way, each node can use a similar algorithm as in the centralized environment to perform backup LSP computations since it has a complete knowledge of the required information. Such solution allows the optimization of the protection bandwidth but it overloads the network with large and frequent messages transmitting the protection costs. With a similar technique, [9] proposes to advertise within IGP-TE protocols the structures and properties of backup LSPs instead of the protection costs. The size and the number of messages transmitted in the network are noticeably decreased with the use of the facility backup protection [11] but they remain high and awkward in large networks.

In order to decrease the quantity of information advertised in the network, [16] proposes the Path Computation Element (PCE)-based MPLS-TE fast reroute technique. With this last technique, a separate PCE is associated with each failure risk in order to compute backup LSPs which will be activated at the failure of that risk. For a failure risk of type node (resp. unidirectional link), the PCE is implemented on the node itself (on the outgoing node to that link). With the knowledge of the link protection capacities transmitted within the Interior Gateway Protocol-Traffic Engineering (IGP-TE), the PCE can select the links which can be used to compute the backup LSPs protecting against the failure of its associated node and/or links. Indeed, since all the computations of backup LSPs protecting against one risk are performed by a same PCE, this last one can determine all the protection costs associated to that risk. Thus, the links that can be used in the backup LSP computation will be deduced accordingly to (4). This computation technique does not require any communication between PCEs but it introduces new constraints limiting its utilization. Firstly, with this technique, non disjoint SRLGs must be managed by a same PCE. This concentrates the computations on some PCEs and can induce identical problems as that encountered in centralized environments. Secondly, with the actual specification of the PCE-based MPLS-TE fast reroute approach, it is not possible to rely on a single NNHOP backup LSP to protect against the failure of a node and its upstream link. Indeed, the PCE which computes backup LSPs protecting against the failure of a link *u*-*v* appearing in a SRLG may be different and far from those (PCEs) which compute backup LSPs protecting against the end nodes of link *u*-*v* (i.e. the node supporting the first PCE is not the same as *u* or *v*). As a result, without bandwidth information exchange between PCEs, the protection against the failure of the node *u* (or node *v*) and its upstream link *u*-*v* requires the use of two backup LSPs: one NNHOP backup LSP protecting against the node failure (*u* or *v*) and one another NHOP backup LSP protecting against the link failure (*u*-*v*). Hence, nodes must be able to distinguish node from link failures to activate the adequate LSPs (ie. LSPs dealing with the failure). This can be done by the use of a double hello mechanism [15] which has the disadvantage of

[3] $FR\ (b)$ corresponds to a set of Failure Risks (*FR*) associated with the backup LSP b. It contains all the risks whose failure activates b.

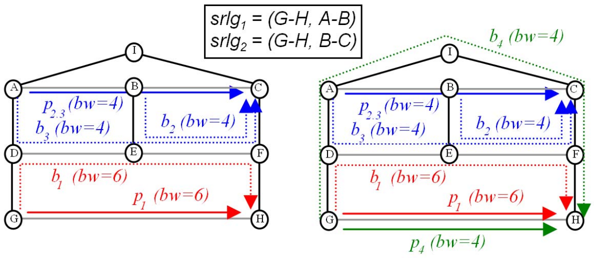
increasing the recovery cycle [14].

To get around the disadvantages quoted above, Kini proposed a new heuristic in [2] to compute the backup LSPs. In this heuristic, the protection cost δ_r^λ of a risk r on an arc λ is approximated by the maximum of the protection costs (G_λ) on that arc. Thus, a new backup LSP of bandwidth bw is computed on the network topology restricted to the unidirectional links λ_i verifying $bw \leq R_{\lambda_i}$ (R_{λ_i} is the residual bandwidth on the unidirectional link λ_i. It corresponds to: $R_{\lambda_i} = RP_{\lambda_i} + RB_{\lambda_i}$). After each computation of a new backup LSP, bandwidth sharing is accomplished by nodes performing the admission control and the new values of protection bandwidth ($\{(R_{\lambda_i}, G_{\lambda_i})\}_{\lambda_i}$) are flooded in the network. Despite of its simplicity, this heuristic presents a very high blocking probability (ie. the number of backup LSPs that can be built with this heuristic is low) and does not bound the quantity of protection bandwidth allocated on arcs. An improvement to the Kini's heuristic can be achieved by dividing the available bandwidth on links in two pools (as described in section 2). In such case, a unidirectional link λ is used in the computation of a new backup LSP of bandwidth *bw* and protecting against the failure risk *r* if $Min(F_r, G_\lambda) + bw \leq BC_\lambda$ (F_r is the cumulative bandwidth of primary LSPs traversing the risk *r*).

4. Targeted Distribution of Resource Allocation (TDRA) algorithm

In the TDRA algorithm, one $BPCE_n$ (Backup Path Computation Element of node *n*) is associated with each node *n*. This $BPCE_n$ is responsible of the computation of backup LSPs protecting against the failures of the node *n* and its incoming adjacent links. As explained in [16], such deployment of BPCEs allows the computation of backup LSPs protecting against risks of type node and/or link. Indeed, each $BPCE_n$ can memorize the structures and properties of all the LSPs that it computes (ie. LSPs protecting against the failure of the node *n* and/or against the failure of adjacent links to *n*). Thus, each $BPCE_n$ can easily deduce the protection costs of the risks *n* and its adjacent links. Obviously, such protection costs allow the computation of backup LSPs protecting against the failure of node *n* and/or the failure of adjacent links to *n* which are not in SRLGs (inequality (4)).

However, when the link to be protected appears in one or several SRLGs, additional information must be communicated to the BPCEs running on the end nodes of that link. This information must allow the computation of the maximum protection costs of the SRLGs including the protected link (inequality (4)). An easy way to determine such maximums consists either to centralize computations of LSPs protecting against non disjoint SRLGs in new BPCEs (as in [16]) or to flood all the SRLG protection costs over the network. These two solutions can be applied for small networks where the number of SRLGs is low.



(a) Situation before the establishment of p_4 and b_4

(b) Situation after the establishment of p_4 and b_4

Figure 2. Backup LSPs computed with TDRA algorithm

In large networks containing a great number of SRLGs, both centralized computations and SRLG protection cost flooding have severe consequences on the network scalability. Indeed, the centralized computations present the risk of bottleneck around the server and increases the recovery cycle whereas the flooding of a great number of SRLG protection costs in large networks increases significantly the network load. To get around these problems and to offer scalability, we propose to use the TDRA algorithm which decreases the quantity of information transmitted in the network by targeting the nodes to be advertised (targeting the recipient nodes). Typically, only the structures and properties of backup LSPs, which can be activated simultaneously after a failure of a SRLG, are shared and transmitted to the end nodes of links appearing in a same SRLG.

To facilitate the understanding of our TDRA algorithm, we illustrate its operation by an example. In figure 2 (a), two NHOP backup LSPs b_1 (*G*→*D*→*E*→*F*→*H*) and b_2 (*B*→*E*→*F*→*C*) and one NNHOP backup LSP b_3 (*A*→*D*→*E*→*F*→*C*) are established to protect the primary LSPs p_1 (*G*→*H*) and $p_{2.3}$ (*A*→*B*→*C*). Thus, b_1 protects the primary LSP p_1 against the failure of link *G-H*, b_2 protects the primary LSP $p_{2.3}$ against the failure of link *B-C* and b_3

	$A \rightarrow D$	$B \rightarrow E$	$D \rightarrow E$	$E \rightarrow F$	$F \rightarrow C$	$G \rightarrow D$	$F \rightarrow H$	$A \rightarrow B$	$B \rightarrow C$	$G \rightarrow H$	others
$\delta^{\cdot}_{H}$	0	0	0	0	0	0	∞	0	0	∞	0
$\delta^{\cdot}_{GH}$	0	0	6	6	0	6	6	∞	∞	∞	0
$\delta^{\cdot}_{FH}$	0	0	0	0	0	0	∞	0	0	0	0
$\delta^{\cdot}_{AB}$	4	0	4	4	4	0	0	∞	0	∞	0
$\delta^{\cdot}_{BC}$	0	4	0	4	4	0	0	0	∞	∞	0

Table 1. Protection costs on node *H*

is used to protect the primary LSP $p_{2.3}$ against the failures of node *B* and link *A-B*.

To accelerate the computation and to ensure the respect of bandwidth constraints, each $BPCE_n$ maintains, in a table, the protection costs of risks including node *n* and its adjacent links. These protection costs are deduced from the structures and properties of backup LSPs received or computed by $BPCE_n$. For instance, the different protection costs maintained by $BPCE_H$ are shown in table 1. This table includes the protection costs of the failure risks *H*, *G-H*, *F-H*, *A-B*, and *B-C* (on the network unidirectional links). These failure risks are classed in two sets: *set of local risks* and *set of distant risks*. The set of local risks of a node *n* includes the node risk *n* and its adjacent link risks. The set of distant risks of a node *n* is composed of the remaining link risks which appear in SRLGs including an adjacent link to the node *n*. For node *H* in figure 2 (a), the set of local risks includes the node *H* itself and its adjacent links *G-H* and *F-H*. The set of distant risks is composed of links *A-B* (*A-B* is in $srlg_1$ which includes the adjacent link *G-H* to the node *H*) and *B-C* (*B-C* is in $srlg_2$ which includes the adjacent link *G-H* to the node *H*).

At the reception of a backup LSP computation request by $BPCE_n$, this last element runs algorithm 1. Concretely, when $BPCE_H$ receives a request to compute a backup LSP b_4 of bandwidth $bw(b_4) = 4$ and protecting the primary LSP p_4 (*G*→*H*) against the failure link *G-H* (figure 2 (b)), it determines initially the set $FR(b_4)$. This set is composed of failure risks including the protected link and the protected node ($FR(b_4) = \{G\text{-}H, srlg_1, srlg_2\}$). After that, $BPCE_H$ deduces the protection costs of the risks belonging to $FR(b_4)$ (table 2). For risks of type node or link, no additional treatment is required since the protection costs of such risks are maintained and directly accessible in the protection cost table of $BPCE_H$ (the line 3 of table 1 is copied into the line 1 of table 2). For risks of type SRLG, (1) is used to deduce their protection costs. Hence, the protection costs of $srlg_1$ (resp. $srlg_2$) are deduced, by adding the protection costs of risks *G-H* and *A-B* (resp. risks *G-H* and *B-C*), and they are copied in the line 2 (resp. line 3) of table 2. Once the protection costs determined, node *H* applies (4) to select the links which can be used in the computation of b_4. Typically, if we consider that the links in figure 2 are of same protection capacity (equal to 10 units), the (unidirectional) links *D*→*E*, *E*→*F*, *A*→*B*, *B*→*C* and *G*→*H* are pruned from the network topology before the computation of the backup LSP b_4 (gray columns in table 2).

In the second step of algorithm 1, $BPCE_H$ searches for a path interconnecting, in the reduced network topology (and thus verifying the bandwidth constraints), node *G* (the PLR node which is the ingress node of b_4) to node *H*. It then determines the unique backup LSP b_4 *G*→*D*→*A*→*I*→*C*→*F*→*H* (figure 2 (b)) which protects p_4 against the failure of *G-H* and satisfies the bandwidth constraints.

	$A \rightarrow D$	$B \rightarrow E$	$D \rightarrow E$	$E \rightarrow F$	$F \rightarrow C$	$G \rightarrow D$	$F \rightarrow H$	$A \rightarrow B$	$B \rightarrow C$	$G \rightarrow H$	others
$\delta^{\cdot}_{GH}$	0	0	6	6	0	6	6	∞	∞	∞	0
$\delta^{\cdot}_{srlg_1}$	4	0	10	10	4	6	6	∞	∞	∞	0
$\delta^{\cdot}_{srlg_2}$	0	4	6	10	4	6	6	∞	∞	∞	0
Max^a	4	4	$\underline{10}$	$\underline{10}$	4	6	6	$\underline{\infty}$	$\underline{\infty}$	$\underline{\infty}$	0
$BC.-bw$	6	6	6	6	6	6	6	6	6	6	6

[a] $Max = Max_{r\in\{G-H,srlg_1,srlg_2\}}\,(\delta^{\cdot}_{r})$

Table 2. Determination of the links which verify the bandwidth constraints

In the third step of algorithm 1, $BPCE_H$ updates its protection cost table by adding the bandwidth quantity of b_4 ($bw(b_4) = 4$) to the protection costs of the risk *G-H* on all the links of b_4. Then, node *H* sends to the end nodes of links appearing in $srlg_1$ and $srlg_2$ (SRLGs belonging to $FR(b_4)$) the structure and properties of the determined backup LSP b_4.

In the last step of algorithm 1, the backup LSP b_4, determined in step 3, is returned.

Algorithm 1 Computation of a backup path *b* verifying bandwidth constraints on a graph *G = (V, E)*

```
Step 1:
$E' = E$
for each arc $\lambda \in E$ do
  $max\_protection\_cost_\lambda = 0$
  for each risk $r \in$ FR (b) do
    $\delta_r^\lambda = \textbf{deduce\_protection\_cost}(\lambda, r)$
    if $max\_protection\_cost_\lambda < \delta_r^\lambda$ then
      $max\_protection\_cost_\lambda = \delta_r^\lambda$
    end if
  end for
  if $max\_protection\_cost_\lambda > BC_\lambda -$ bw (b) then
    $E' = E' \setminus \{\lambda\}$
  end if
end for
Step 2:
$b = \textbf{bypass\_path}(\textbf{PLR}(b), G'(V, E'), \textbf{primary\_path}(b))$
{PLR (b) returns the source node of b and primary_path (b) returns the primary path protected by b}
Step 3:
update_and_send_protection_costs (FR (b), bw (b))
Step 4:
return b
```

5. Protocol extensions

To compute the backup LSPs with TDRA algorithm, some extensions and/or modifications to the existing protocols are required/desired.

Firstly, it is necessary to configure and signal the different SRLGs to the network nodes (LSRs). For that, the IGP-TE protocol extensions described in [4, 5] can be adopted.

Secondly, to exchange and share the backup LSP structures and properties between all the end nodes of links appearing in a same SRLG, we propose to extend the signaling protocols. Here (section 5.1), we focus on the RSVP-TE protocol extensions [1]. Similar extensions can be applied to the other signaling protocols.

Finally, we propose in section 5.2 some slight extensions to the IGP-TE protocols (OSPF-TE [5] and IS-IS [4]) in order to advertise the protection capacities.

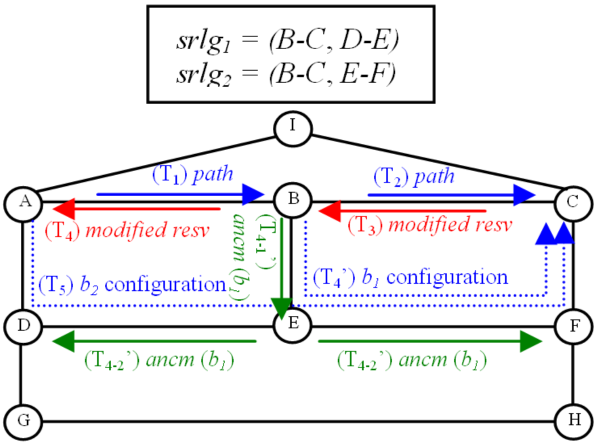

$srlg_1$ = (B-C, D-E)
$srlg_2$ = (B-C, E-F)
I
A
B
C
D
E
F
G
H
(T_1) *path*
(T_2) *path*
(T_4) *modified resv*
(T_3) *modified resv*
(T_5) b_2 configuration
(T_4') b_1 configuration
(T_{4-1}') *ancm* (b_1)
(T_{4-2}') *ancm* (b_1)
(T_{4-2}') *ancm* (b_1)

Figure 3. Message sequencing for the setup of a primary protected LSP ($A \rightarrow B \rightarrow C$)

5.1. RSVP-TE extensions

With the RSVP-TE extensions introduced in [11], the Head-End LSR (the source LSR of the primary LSP) can ask for primary LSP protection by setting the flag "local protection desired" and/or by including in the RSVP-TE *path* message the FAST_REROUTE object. In order to allow the computation of backup LSPs with our TDRA algorithm, we propose to define a new object BACKUP_LSP and a new type of message called *ancm* (or *announcement*). The BACKUP_LSP object transports the backup LSP structure and properties. It is conveyed in *path*, *resv* and *ancm* messages. The message *ancm* is used to transmit the BACKUP_LSP object to the End Nodes of Links appearing in SRLGs (ENLS) which include the protected link.

When the destination node *dest* of a primary LSP receives a *path* message asking for protection, it computes initially a backup LSP protecting against the failure of its incoming link ($BPCE_{dest}$ is used for this computation). Then, it builds a BACKUP_LSP object including the backup LSP structure and properties (identifier, bandwidth, protected link, type of the backup LSP, explicit route, etc.). After this step, the node *dest* constructs a (*modified*) *resv* message, in which it inserts the BACKUP_LSP object, and sends it to its upstream node on the primary LSP. At this time, the node *dest* creates a new *reservation state block* for the primary LSP (as specified in [1]) and constructs a new *backup state block* for the backup LSP. This last state is kept into the node *dest* until the backup LSP is deleted or until the protected link fails.

When a node of the primary LSP receives a *resv* message conveying a BACKUP_LSP object, it[4] makes same treatments as that performed by the node *dest* with the difference that the computed backup path is a NNHOP LSP. In addition to this processing, each node of the primary LSP, which receives a *modified resv* message, extracts the BACKUP_LSP object and configures the corresponding backup LSP.

When the protected link belongs to a group of SRLGs, additional treatments are required to share the backup LSP structure and properties. Typically, each node PLR setting a backup LSP, which protects against the failure of a link appearing in SRLGs, informs all the ENLS (except its downstream primary node) of the establishment of this new backup LSP. Thus, the PLR builds an *ancm* message, in which it inserts the BACKUP_LSP object computed and transmitted by its downstream primary node, and sends[5] it to the ENLS. We note that these *ancm* messages are sent directly to the ENLS by specifying their IP addresses in the IP header (without *Router Alert* option). Besides, each node receiving an *ancm* message creates a new *backup state block* for the backup LSP included in the BACKUP_LSP object. In a same manner and to inform the downstream primary node of the success (resp. the failure) of backup LSP configuration, the PLR node includes (resp. does not include) the BACKUP_LSP object in the next *path* messages refreshing the primary LSP.

5.1.1 Enhancement of the ancm message distribution

To decrease the number of messages transmitted in the network, we suggest the use of a KMB tree [7], which covers all the ENLS, to route the *ancm* messages. Such tree is an approached Steiner tree in which all the children nodes

[4] The source node of the primary LSP does not compute any backup LSP and it does not forward the *modified resv* message.

[5] For performance and implementation considerations, the *ancm* messages are sent by the PLRs instead of the nodes supporting the BPCEs.

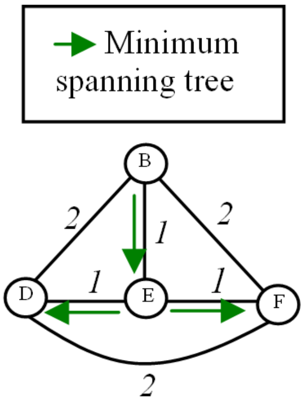


Figure 4. Minimum spanning tree covering all the nodes of the transitive graph

are connected to their parent node with the use of shortest paths. It is computed on the transitive graph which includes only the ENLS. We note that, each couple of nodes, in a transitive graph, is interconnected directly through a link whose cost is equal to the distance (number of hops) between the two (couple's) nodes. After the determination of the transitive graph, we deduce a minimum spanning tree (MST) [8] covering all the nodes of the transitive graph. This MST defines the KMB tree used for routing and as a result, it specifies the paths that the *ancm* messages will follow. Typically, each MST node sends one *ancm* message to each MST child.

To avoid the transmission of tree structures in ancm messages, nodes compute KMB trees covering the end nodes of links which appear in SRLGs including an adjacent edge, at each discovery of a new SRLG in the network. A tie-breaking rule is used to ensure that the trees covering a same set of nodes and computed on different nodes will be identical.

5.1.2 Example of message exchanges for the establishment of a primary protected LSP

In figure 3, node *A* receives a request to establish a primary protected LSP interconnecting node *A* to node *C*. At this time, node *A* determines the lowest cost path *A*→*B*→*C* and sends a *path* message to its next hop (node *B*) in order to configure this path (at time T_1). When node *B* receives the *path* message, it creates a *path state block* for the new primary LSP and sends a *path* message to node *C* at time T_2 ($T_2 > T_1$). When a *path* message arrives to the destination node of the primary LSP (node *C*), this last node analyses the message and deduces that the primary LSP requires a protection. Thus, node *C* computes a backup LSP b_1 (*B*→*E*→*F*→*C*) which protects against the failure of its upstream link *B*→*C* and builds a BACKUP_LSP object specifying that path. After the admission control performed on link *B*→*C*, node *C* builds a *modified resv* message including the BACKUP_LSP object and sends it to node *B* at time T_3.

When node *B* receives the *modified resv* message sent by node *C*, it extracts from this message the BACKUP_LSP object and configures the backup LSP b_1 at time $T_4{}' > T_3$.

As the protected link *B*→*C* is in two SRLGs ($srlg_1$ and $srlg_2$), node *B* (the upstream node to *C*) informs the ENLS (ie. *B*, *D*, *E* and *F*) of the establishment of the backup LSP b_1. Thus, it determines initially the transitive graph which includes the end nodes (*B*, *D*, *E* and *F*) of links appearing in $srlg_1$ and $srlg_2$. For instance, if we consider that all the links of the network topology shown in figure 3 are of equal cost then the transitive graph which includes nodes *B*, *D*, *E* and *F* is illustrated in figure 4. On this graph, node *B* deduces the MST (which covers the four nodes *B*, *D*, *E* and *F*) and sends an *ancm* message to its unique MST at time $T_{4-1}{}'$ ($T_{4-1}{}' > T_4{}'$). In its turn, node *E* receives the *ancm* message, treats it and redirects it to its two MST children *D* and *F* (at time $T_{4-2}{}' > T_{4-1}{}'$). When nodes *D* and *F* receive the *ancm* messages sent by *E*, they delete them since they don't have children in the MST.

In parallel to the previous treatment, node *B* performs the admission control on link *A*→*B*, compute a NNHOP backup LSP (b_1) protecting against the failure of node *B* (and against the failure of link *A*-*B*) and sends a *modified resv* message to node *A* at time $T_4 > T_3$.

In the last configuration step, node *A* receives the *modified resv* message and treats it. Hence, node *A* extracts from the message the BACKUP_LSP object and configures the backup LSP b_2 at time $T_5 > T_4$.

To keep trace of the valid primary and backup LSPs, each RSVP-TE node refreshes the LSPs traversing it. This is done by sending a *path* message (resp. a *resv* message) to the next hop (resp. to the preceding node) at each period of time. In our extensions to RSVP-TE, we propose also that nodes retransmit the *ancm* messages at each period of time. Besides, to indicate that the backup LSP configuration succeeds (resp. fails), nodes should include (resp. exclude) the received BACKUP_LSP object in the next *path* messages refreshing a primary protected LSP. In figure 3 for instance, node *A* (resp. node *B*) includes the BACKUP_LSP object associated with the backup LSP b_2 (resp. with b_1) in all the *path* messages refreshing the primary protected LSP and sent after the configuration success of the backup LSP b_2 (resp. b_1).

5.2. IGP-TE extensions

To announce the link protection capacities, we propose to use the TE parameters defined in OSPF-TE [5] and in

ISIS-TE [4]. Typically, we suggest defining a new link sub-TLV (4 bytes) to advertise the protection capacity of each unidirectional link. Such sub-TLV will be carried within the OSPF Link TLV and the IS-IS Extended IS reachability TLV.

6. Analysis and simulation results

6.1. Simulation model

In order to evaluate the performances of the TDRA algorithm, we compare it to the Improved Kini Heuristic (IKH heuristic) described in the end of section 3. Two metrics are used for this purpose: ratio of rejected backup LSPs (RRL) and mean number of messages (MNM) transmitted in the network per configured backup LSP. The first metric measures the ratio of backup LSPs that are rejected because of the lack of protection bandwidth on the network links. It corresponds to the ratio between the number of backup LSP requests that are rejected and the total number of backup LSP requests. The second metric counts the mean number of messages traversing the network links, after each backup LSP establishment, to maintain and update the protection bandwidth information necessary for computation. Formally:

RRL = # rejected protection requests / # protection requests

$MNM = \sum_{\lambda \in E}$ *# messages traversing* (λ)

/ # accepted protection requests

Where E is the set of network unidirectional links.

The network topology used in our tests, with 15 nodes and 56 unidirectional links, is shown in figure 5 (each line in the figure represents two opposite unidirectional links). The available bandwidth on the network links is divided in two pools: primary pool and protection pool. The primary pool capacity of links is assumed infinite (i.e. the primary pool capacities are sufficient to satisfy all the requests of primary path establishment) whereas the protection capacity is equal to 100 units on the light links and equal to 400 units on the bold links. In this network topology, we created 16 SRLGs (represented by ellipses in figure 5) to observe the effect of SRLG presence on the backup LSP computation techniques compared here.

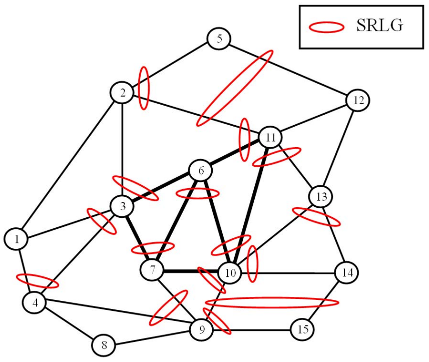


Figure 5. Test network

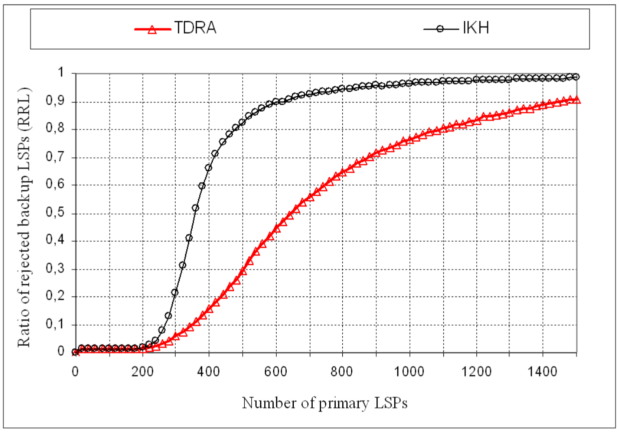


Figure 6. Ratio of rejected backup LSPs (RRL)

The traffic matrix is generated randomly and consists of LSPs asking for quantities of bandwidth uniformly distributed between 1 and 10. The LSPs are computed with the use of the Dijkstra's shortest path algorithm. Their ingress and egress nodes are chosen randomly among the nodes of the network.

At each establishment of 20 primary LSPs, the two metrics RRL and NMN are computed for the two compared methods. We note that our results correspond to metric mean values of 1000 experiments.

6.2. Results and analysis

Figure 6 depicts the evolution of RRL as a function of the number of primary LSPs setup in the network. In this figure, we observe that the RRL values of the TDRA algorithm are lower (or equal) and better than those of the IKH heuristic. This is due to the complete knowledge of the required protection bandwidth information (protection costs) with the TDRA algorithm whereas the IKH heuristic uses only partial information (maximal protection costs on links) to compute the backup LSPs.

Typically, when the number of primary LSPs is lower than 220 (low network loads), both the TDRA algorithm and the IKH heuristic have very comparable RRL values. Besides, these RRL values are very close to zero. This can be explained by the protection cost values on links which are very small for low network loads. Hence, nearly all the

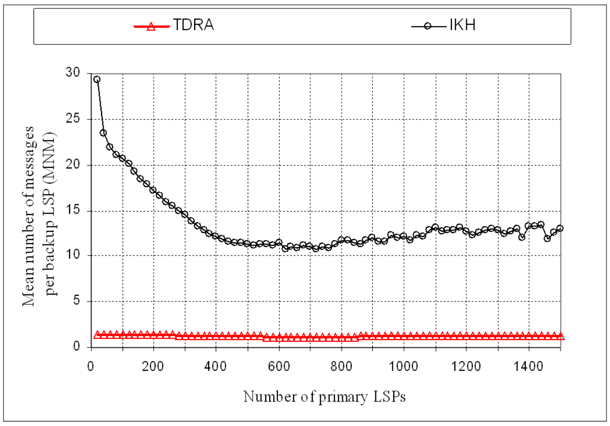


Figure 7. Mean number of messages sent in the network per backup LSP

protection requests are satisfied.

When the number of primary LSPs is higher than 220, the difference between the RRL values of the TDRA algorithm and the IKH heuristic becomes apparent and high. This is due to the overestimation of the protection costs with the IKH heuristic whereas the TDRA algorithm uses exact values of protection costs. In fact, with the increase of the network load, the maximal protection costs on links attain rapidly high values (values higher than the protection capacity minus the maximal bandwidth of LSPs). Thus, approximating the protection costs of each risk by the maximal protection cost (on the network links), as in the IKH heuristic, causes the reject by mistake of a great number of links before the step of backup LSP computation (step 2 of algorithm 1). With the TDRA algorithm however, no approximation is needed: the values of the required protection costs are known by the BPCEs. As a result, the backup path computations are performed more efficiently.

With regards to the second metric, figure 7 shows that the mean number of messages transmitted in the network with the TDRA algorithm is very lower and better than that obtained with the IKH heuristic. Contrarily to the IKH heuristic which broadcasts systematically the new values of protection bandwidth, the TDRA algorithm decreases the mean number of messages sent in the network by the selection of nodes to be advertised: only the end nodes of links appearing in SRLGs including the protected link are informed of the structure and properties of the new backup LSPs.

In figure 7, we see that the MNM values of the TDRA algorithm are almost constant (very slight diminutions) and vary between 1.36 and 1.21. As long as the protection requests are satisfied, the MNM should be constant. Indeed, in such case, the MNM depends only on the SRLG structures. However, in our simulation, the MNM decreases slightly because the links appearing in SRLGs are overloaded more quickly than those which do not (the protection of a link, which does not appear in any SRLG, does not require any message transmission).

Concerning the MNM of the IKH heuristic, figure 7 shows that its values are higher for small network loads where the maximum protection costs on links change more quickly. After the establishment of the first 400 primary LSPs, the mean number of messages sent in the network with the IKH heuristic seems to be stabilized in the surrounding of 12 messages per established backup LSP (i.e. approximately 0.8 broadcasts per backup LSP). This is due to the increase of the maximum protection costs on links which reach high values changing less quickly.

7. Conclusion

In this article, we proposed a Targeted Distribution of Resource Allocation (TDRA) algorithm, to compute online the backup LSPs. Our algorithm shares effectively the bandwidth between backup LSPs which protect against all the types of failure risk. It is also scalable and balances equitably the computation task on the network nodes. By targeting the nodes to be notified at each backup LSP setup, the TDRA algorithm decreases significantly the number of messages sent in the network (to allow the backup LSP computation). Besides, the TDRA algorithm is easy to be deployed and requires only slight extensions to the signaling and routing protocols.

Simulation results show that the TDRA algorithm improves notably the bandwidth sharing by decreasing the number of rejected backup LSPs. Moreover, the TDRA algorithm decreases the number of messages transmitted in the network to accomplish the backup LSP computation.

References

[1] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, December 2001.

[2] S. Kini, K. Kodialam, T. V. Lakshman, S. Sengupta, and C. Villamizar. Shared Backup Label Switched Path Restoration. Internet Draft draft-kini-restoration-shared-backup-01.txt, IETF, May 2001.

[3] M. S. Kodialam and T. V. Lakshman. Dynamic Routing of Restorable Bandwidth-Guaranteed Tunnels using Aggregated Network Resource Usage Information. *IEEE/ACM Transactions On Networking*, 11(3):399–410, June 2003.

[4] K. Kompella and Y. Rekhter. Intermediate System to Intermediate System (IS-IS) Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS). RFC 4205, October 2005.

[5] K. Kompella and Y. Rekhter. Ospf Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS). RFC 4203, October 2005.

[6] K. Kompella and Y. Rekhter. Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS). RFC 4202, October 2005.

[7] L. Kou, G. Markowsky, and L. Berman. A Fast Algorithm for Steiner Trees. *Acta Informatica*, 15:141–145, June 1981.

[8] J. B. Kruskal. On the Shortest Spanning Subtree and the Traveling Salesman Problem. In *Proceedings of the American Mathematical Society*, volume 7, pages 48–50, 1956.

[9] J. L. Le Roux and G. Calvignac. A Method for an Optimized Online Placement of MPLS Bypass Tunnels. Internet Draft draft-leroux-mpls-bypass-placement-00.txt, IETF, February 2002.

[10] P. Meyer, S. Van Den Bosch, and N. Degrande. High Availability in MPLS-based Networks. Alcatel telecommunication review, Alcatel, 4th Quarter 2004.

[11] P. Pan, G. Swallow, and A. Atlas. Fast Reroute Extensions to RSVP-TE for LSP Tunnels. RFC 4090, May 2005.

[12] S. Ramamurthy and B. Mukherjee. Survivable WDM Mesh Networks (Part 1 - Protection). *In IEEE INFOCOM*, 2:744–751, 1999.

[13] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, January 2001.

[14] V. Sharma and F. Hellstrand. Framework for Multi-Protocol Label Switching (MPLS)-based Recovery. RFC 3469, February 2003.

[15] J. P. Vasseur and A. Charny. Distinguish a Link from a Node failure using RSVP Hello Extensions. Internet Draft draft-vasseur-mpls-linknode-failure-00.txt, IETF, November 2002.

[16] J. P. Vasseur, A. Charny, F. Le Faucheur, J. Achirica, and J. L. Le Roux. Framework for PCE-based MPLS-TE Fast Reroute Backup Path Computation. Internet Draft draft-leroux-pce-backup-comp-frwk-00.txt, IETF, July 2004.