Improved dual-forest for multicast protection

Mohand Yazid Saidi, Bernard Cousin, Miklos Molnar

Improved Dual-Forest for Multicast Protection

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

Abstract-- **To provide fault-tolerance for multicast connections, different techniques of protection are developed. These techniques can be classed into reactive and pro-active approaches. Reactive approaches can have long recovery latency which is undesirable for many types of applications such as the real time ones.**

In this paper, we focus on the multicast pro-active fault-tolerance schemes. One of the promising protection techniques is the dual-tree protection which is efficient to cope with single link failure but which cannot deal with node failures suitably. In this paper, we present an improved solution using a dual-forest for multicast protection. Our proposition provides three improvements to traditional dual-tree protection. The first one concerns the capability to bypass both single link and node failures in a suitable and quick manner. The second increases the level of protection with the use of a forest as backup instead of a tree. The last permits the cost optimization of the dual-forest and of the delivery tree after recovery. Simulation experiments show that the improved dual-forest scheme has better protection rate and causes less tree cost increase after recovery than the path-protection scheme.

Keywords-- **network, multicast, protection, dual-tree, dual-forest**

I. INTRODUCTION

Due to the augmentation of delay sensitive network applications, fault-tolerance techniques become very important and necessary in the domain of routing. These techniques aim to ensure the non-interruption or minimal disruption of communications in a cost efficient manner (without traffic duplication). Thus, when a failure is detected, the protection techniques must determine quickly backup paths which will be used to bypass the failed component.

Existing protection schemes can be classed in two categories: reactive and pro-active. With reactive approaches, no computation of backup paths is needed before the failure. The restoration consists in finding and configuring new paths allowing the restoration of communications after the failure detection. However, in the pro-active approaches, the backup paths are pre-computed and possibly pre-configured beforehand. When a failure occurs, the protection mechanism switches from the primary affected paths to their backups.

The first type of protection (reactive) works well enough for datagram communications and has the advantages of flexibility to cope with topology changes and decreasing the computational and maintenance costs. However, recovery latency measured using such type of protection is long and undesirable for many types of communications. In high-speed networks, long recovery from failure is very awkward and causes the loss of much data. Moreover and since efficient pre-reservation of bandwidth is not possible in the reactive protection techniques, the recovery can fail because the amount of available bandwidth on backup paths is not sufficient to receive the traffic of their affected primary paths.

To get around the previous problems, the tendency is actually to envisage the use of pro-active protection schemes. In such type of protection, the recovery is faster because the backup paths are pre-computed and generally pre-configured. Moreover, the pre-configuration of backup paths ensures the sufficiency of bandwidth and the success of the recovery.

Many pro-active techniques are developed for unicasting. In end-to-end protection, the whole path between the source and the destination is protected by only one vertex-disjoint backup path [1, 2]. In one-to-one protection, the primary path is divided into a set of segments; each one has its own backup path and can be as small as a link [1, 3].

For multicast, the routing structure to protect is a tree. Protection in this case is more challenging to achieve than in the unicast case since one network failure affects all members downstream the failing component in the multicast tree.

The first trivial proposition for multicast pro-active protection scheme suggests to extract from the multicast tree all the paths from the root to each member and to protect each path with a unicast pro-active protection scheme [4, 5]. This proposition has several drawbacks: it does not guarantee neither the non-duplication of the traffic over links nor the non-formation of loops after recovery.

In recent works, the tendency is to search a routing structure which is capable to protect all nodes and links of the primary tree. In [6], a dual-tree scheme for multicast fault-tolerant is proposed. In this protection technique, a new tree is built by the interconnection of the primary tree leafs without the use of any link or inner node of the primary tree. This tree provides paths which can be used to bypass failures. As we will see in the next section, the proposed dual-tree scheme can protect only against single link failure. In this paper, we propose the dual-forest protection scheme which uses a forest for protection. This forest interconnects the maximum number of primary tree leafs without the use of any link or inner node of the primary tree. Our solution makes several improvements to the dual-tree protection scheme. Indeed, it can cope with both link and node failures successfully, increases the level of protection and optimizes both the cost of delivery tree after recovery and the cost of backup paths (dual-forest).

The rest of this article is organized as follows. In section II, we review works related to multicast pro-active protection schemes and outline their insufficiencies. In section III, we describe our proposition which improves dual-tree scheme. Simulation results are presented and discussed in section IV. The last section is dedicated to the conclusions.

II. Related Works

Pro-active protection techniques for multicast can be grouped in two categories: global level and local level [1].

In the global level protection techniques, the switching from primary to backup paths when a failure is detected is done by the source node. As a result, the failure notification message must reach the source node in order to start recovery procedure. In the case of link or node failure far from the source node, the delay may be significant and undesirable. However, in local level protection techniques, the recovery is faster because the activation of backup paths is triggered by the node detecting the failure (or by one of its upstream nodes) which is nearer to the failed component.

In all the following sections, the network topology is represented by a directed graph *G*. The set of destination nodes *Mi* corresponds to the set of multicast group members which is noted *M*. The source node and the primary tree are noted *S* and T_p respectively.

Due to the lack of space, we present here only two techniques of protection: path protection and dual-tree protection. Other techniques can be found in [1, 3, 7].

A. Path protection

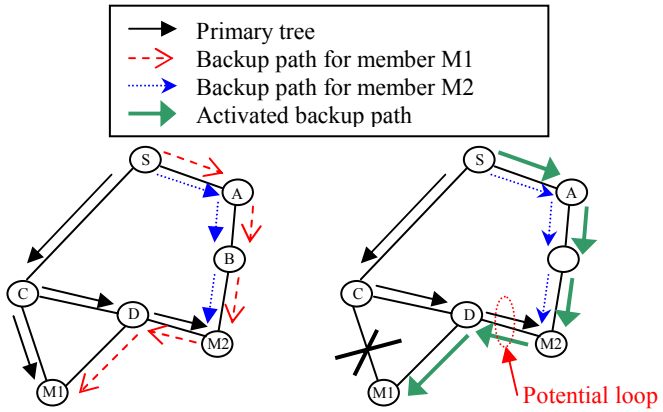
The path protection is a global protection technique inspired from unicast. For each path of the primary tree from the source to a multicast group member, a vertex-disjoint path connecting the source to the member is pre-computed and possibly set up as backup [1, 2]. In Fig. 1(a), the primary path (S, C, M1) (resp. (S, C, D, M2)) connecting the source node to the multicast group member *M1* (resp. *M2*) is protected by the vertex-disjoint backup path (S, A, B, M2, D, M1) (resp. (S, A, B, M2)).

When a failure is detected, the source node which is informed determines all the affected members (or members belonging to the sub-tree newly disconnected from the source node) and activates their backup paths by sending traffic on the (residual) primary tree and on the activated backup paths.

In Fig. 1(b), after the failure detection of link (C, M1), *S* concludes that only the member node *M1* is affected and activates its backup path (S, A, B, M2, D, M1). However and in order to avoid possible loops resulting from the activation of new backup paths (see link (D, M2) in Fig. 1(b)), the node source should send multicast traffic on each activated path and on the (residual) primary tree with different connection identifiers.

This protection technique is easy to implement but presents several drawbacks:

1. It doesn't guarantee the non-duplication of packets over links (resource wasting).
2. A recovery procedure can imply the reconfiguration of several paths.
3. The cost of backup paths is not optimal.



(a) Situation before the failure **(b) Situation after the recovery from link (C, M1) failure**

Fig. 1. Path disjoint protection

B. Dual-tree protection

The dual-tree protection [6] is a local technique of protection using a (dual-)tree for restoration. This dual-tree is built by the interconnection of all primary tree leafs without using any link or inner node of the primary tree. The source node is regarded as a leaf if it has only one child in the primary tree.

In Fig. 2(a), a primary tree is built to convey multicast traffic to the different members (M1, M2, M3, M4) and a dual-tree is pre-computed to deal with failures. The dual-tree is obtained by the interconnection of the four primary tree leafs *M1*, *M2*, *M3* and *M4* without the use of any link or inner node of the primary tree.

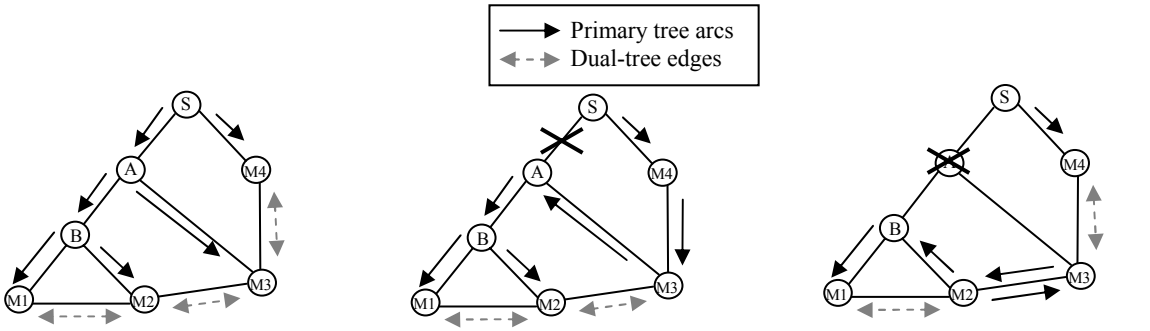
When a node detects a failure on its upstream interface, it runs the restoration algorithm of the dual-tree to repair the multicast communication. In Fig. 2(b), node *A* detects a failure on its upstream interface and starts recovery procedure.

Firstly, it divides the set of primary tree leafs into two sets: affected nodes (AF_A) and unaffected nodes (NA_A). Affected nodes consist in primary leaf nodes which belong to the primary sub-tree rooted at *A* and unaffected nodes correspond to the rest of primary tree leaf nodes. Here:
$AF_A = \{M1, M2, M3\}$, $NA_A = \{M4\}$.

Secondly, node *A* determines one path interconnecting directly (or via intermediary nodes which are not a primary leafs) in the dual tree one node of AF_A to one node of NA_A. The path consisting in link (M3, M4) is returned.

Finally, node *A* sends a *Reconfig* message to node *M3* which will be its new parent. When node *M3* (which is a descendant of *A* in the old primary tree) receives the Reconfig message, it forwards the massage to *M4* and changes its old parent (which is *A*) to be a child. At a reception of Reconfig message by node *M4* (which is an unaffected node), it creates a new multicast state information to serve member node *M3*.

Formally, the specification of the restoration algorithm of dual-tree can be summarized in the following steps:



(a) Situation before any failure

(b) Situation after the recovery from the (S, A) link failure

(c) Erroneous situation after the recovery from the A node failure

Fig. 2. Dual-tree protection

1. When a node x detects a failure on its upstream interface, it computes the two sets AF_x and NA_x.
2. Node x selects one node y in AF_x which is connected to a node z in NA_x using a path p. The inner nodes of p must not belong to ($AF_x \cup NA_x$).
3. Node x sends a Reconfig message to its child leading to y, updates its parent to be that child. Reconfig message will be forwarded in the primary tree until it reaches y, every node receiving this message changes its old parent to be a child, and its old child (on the path to y) becomes its new parent. Reconfig message is then forwarded to z along path p and corresponding multicast information states are installed in the intermediate node(s).

Note that, in dual-tree protection, nodes must know (or deduce) the structures of primary and secondary trees. One way to achieve this is to distribute the topology and group membership information to all the multicast routers. Moreover, all links of the dual-tree must be bi-directional.

The repair of the multicast communication with this protection technique is sure in the link failure cases but it is not guaranteed in the node failure cases. Fig. 2(c) shows a case of node failure where recovery procedure fails to repair multicast communication. In the figure, node *A* fails. As a result, node *M3* and *B* detect a failure on their upstream interfaces and start recovery procedure. Let suppose *M3* is the first node discovering the failure. *M3* computes its AF_{M3} and NA_{M3} sets (AF_{M3} = {M3}, NA_{M3} = {M1, M2, M4}), selects a path (*M3*, *M2*) to get round the failure and sends Reconfig message to node *M2*. Then in same manner, *B* computes its AF_B and NA_B sets (AF_B = {M1, M2}, NA_B = {M3, M4}), chooses the only path (*M2*, *M3*) allowing the interconnection of one node in AF_B to one node in NA_B and sends the Reconfig message to *M3*. The recovery procedure ends with the configurations illustrated in Fig. 2(c) in which members *M1*, *M2* and *M3* are disconnected from the multicast source.

Note that we supposed in the example of Fig. 2 that nodes are not able to differentiate node failures from link failures. This is generally the case in actual networks. However and even if nodes are capable to determine that it is a node failure, the dual-tree protection cannot be sure. Indeed, nodes *M3* and *B* of Fig. 2(c) compute their affected set and unaffected set which are the same ($AF_{M3,B}$ = {M1, M2, M3}, $NA_{M3,B}$ = {M4}) and use the same path (M3, M4) to bypass the failure. As a result, some members (*M1* and *M2* in Fig. 2(a)) will not receive the multicast traffic because they are not connected to the source node.

III. Improved Dual-Forest Protection

In order to protect against both link and node failures, we propose here the improved dual-forest protection scheme. Like the dual-tree protection scheme, the dual-forest scheme uses a *reduced topology* to build the backup paths. This reduced topology is obtained by the elimination in the global topology of all links and inner nodes of the primary tree. Hence and since the reduced topology can be unconnected, we use a forest to best protect the multicast communication.

In sub-section III.A, we present the restoration algorithm of our improved dual-forest protection. Thanks to the modifications made to the algorithm presented in section II.B, this algorithm can cope successfully with both single node and link failures. In sub-section III.B, we ascertain and prove that the KMB forest covering all primary leafs is a good backup structure which optimizes both the cost of backup paths and the multicast tree cost increase after recovery. We define a KMB forest as a forest of KMB trees on each connected component and we point out that a KMB tree [8] is an approached Steiner tree built by the interconnection of the closest Steiner nodes (primary tree leafs) until forming a tree.

A. Restoration algorithm of our improved dual-forest protection

To be able to deal with node failures, we propose some improvements to the algorithm described in section II.B. For simplicity, we present the directed tree model (SSM) of this protection which can be generalized to the undirected tree model (ASM).

When a node x detects a failure on its upstream interface in the primary tree, it divides the set of primary leafs into three sets: surely affected nodes SA_x, possibly affected nodes PA_x and unaffected nodes NA_x.

SA_x consists in leafs of the primary sub-tree rooted at x. PA_x corresponds to leafs of the primary sub-tree rooted at the parent of x and not belonging to SA_x. NA_x is composed of the rest of the primary tree leafs.

Algorithm 1. Improved dual-forest algorithm executed by node *x* detecting a failure

1. **deduce_sets**(x, T_p, SA_x, PA_x, NA_x) ;
   { The node *x* which detects the failure deduces the three sets SA_x, PA_x, NA_x relating to the primary tree T_p }
2. **deduce_backup_path**(F_d, SA_x, PA_x, NA_x, bp_x) ;
   { deduces a shortest path bp_x which belongs to the dual-forest F_d and which interconnects one node in SA_x to one node in NA_x
   if such path does not exist, a shortest path which interconnects one node in SA_x to one node in PA_x is assigned to bp_x
   if no path is determined, an *infinite path* is assigned to bp_x }
3. **if** bp_x = *infinite path* **then goto** end ;
   { recovery with the improved dual-forest protection fails }
   endif
4. **split_backup_path**(PA_x, bp_x) ;
   { if bp_x includes a node of PA_x then a shortest sub-path interconnecting the first extremity node of bp_x (which belongs to SA_x) to the closest node of PA_x is assigned to bp_x
   do nothing if bp_x does not include any node of PA_x }
5. **extremities**(bp_x, e_1, e_2) ;
   { the first extremity of the path bp_x belonging to SA_x is returned in the parameter e_1
   the second extremity of the path bp_x is returned in the parameter e_2 }
6. **create_Reconfig_message**(x, T_p, bp_x, r_msg) ;
   { all nodes on the path in T_p from *x* to extremity e_1 are listed in the *Reconfig* message *r_msg*, in this order
   These nodes will be followed by nodes of bp_x (from e_1 to e_2), in order too }
7. **send_Reconfig_message**(x, r_msg) ;
   { r_msg will be sent to the ***succ(x, r_msg)*** node which is the successor of *x* in the Reconfig message list }
8. **parent**(x, T_p) ← **succ**(x, r_msg) ;
   { the successor of *x* in *r_msg* list becomes its new parent in the delivery tree T_p }
9. **del_child**(x, T_p, **succ**(x, r_msg)) ;
   { the ***succ(x, r_msg)*** node is deleted from the child list of *x* in T_p }

The idea of the restoration algorithm of the improved dual-forest is to give priority to backup paths interconnecting nodes of SA_x to nodes of NA_x. Indeed, these paths are sure and can deal with both node and link failures. We recall that during normal operation, traffic is delivered through the primary tree. The dual-forest which provides the backup paths interconnects primary leafs in the reduced topology. The number of trees forming this dual-forest must be equal to the number of connected components in the reduced topology.

The restoration algorithm of the improved dual-forest works as follows:

Algorithm 2. Improved dual-forest algorithm executed by node *y* receiving a Reconfig message

1. **if** y != e_2 **then**
   { e_2 is the last node in the Reconfig message list }
   send_Reconfig_message(y, r_msg) ;
   { *r_msg* will be sent to the ***succ(y, r_msg)*** node }
   parent(y, T_p) ← **succ**(y, r_msg) ;
   { the parent of *y* in T_p will be the ***succ(y, r_msg)*** node }
   del_child(y, T_p, **succ**(y, r_msg)) ;
   { the ***succ(y, r_msg)*** node is removed from the child list of *y* in T_p }
   endif
2. **add_child**(y, T_p, **pred**(y, r_msg)) ;
   { the predecessor of *y* in *r_msg* list becomes a new child of *y* in T_p }

1. When node *x* of the primary tree detects a failure on its upstream interface, it runs the routine depicted in Algorithm 1.
2. When node *y* receives the Reconfig message, it runs the routine depicted in Algorithm 2.

In addition to the enhancement allowing the recovery from both single link and node failures, the improved dual-forest protection scheme produces other improvements. Firstly, the choice to use a forest instead of a tree as backup increases the protection (see section IV.C.1). Indeed, if it is not possible to determine a tree which interconnects all primary leafs (when the reduced topology is not connected), the use of a forest will allow a protection of some parts of the primary tree. These protected parts consist in the sub-trees whose leafs are completely interconnected by one of the (backup) trees belonging to the dual-forest. Secondly, the restoration delay can be decreased since steps 1 up to 6 of Algorithm 1 can be performed before the detection of a failure. Indeed, each node of the primary tree can suppose the failure of its upstream interface and carry out the required computations to repair the multicast communication.

Let us illustrate the operation of the restoration algorithm of the improved dual-forest by applying it to the network of Fig. 2(a). When node *A* fails, nodes *M3* and *B* will detect the failure on their upstream interfaces. Let suppose that *M3* is the first to start its computations (recovery procedure).

Recovery procedure for M3

As it is described in the restoration algorithm, *M3* will run the routine depicted in Algorithm 1 when it detects a failure:

M3 execution trace of Algorithm 1

1. SA_{M3} ← {M3}, PA_{M3} ← {M1, M2}, NA_{M3} ← {M4}
2. bp_{M3} ← (M3, M4)
   { (M3, M4) is the only path interconnecting one node in SA_{M3} to one node in NA_{M3} }
3. do nothing { because bp_{M3} != *infinite path* }
4. do nothing
   { because bp_{M3} does not include any node of PA_{M3} }
5. e_1 ← M3, e_2 ← M4

6. Reconfig message is created. The list of nodes included in the message is: [M3, M4]
7. Reconfig message is sent to node M4 which is the successor of M3 in the Reconfig message list.
8. parent(M3, T_p) ← M4
   { node *M3* sets *M4* as its parent in T_p }
9. del_child(M3, T_p, M4)
   { do nothing because *M4* is not a child of *M3* in T_p }

Since *M3* sends a Reconfig message to *M4* (in step 7), this latter will receive the message and will run the routine depicted in Algorithm 2 accordingly to the restoration algorithm of the improved dual-forest.

M4 execution trace of Algorithm 2

1. do nothing { because $M4 = e_2$ }
2. add_child(M4, T_p, M3)
   { *M3* becomes a child of *M4* in T_p }

At this point of execution, the *M3* recovery procedure ends with the tree structures shown in Fig. 3(a).

Independently from the time when *M3* starts and ends its recovery procedure, *B* will detect the failure on its upstream interface and starts the computations relating to the recovery procedure in order to cope with the failure.

Recovery procedure for B

Like node *M3*, node *B* will run the routine depicted in Algorithm 1:

B execution trace of Algorithm 1

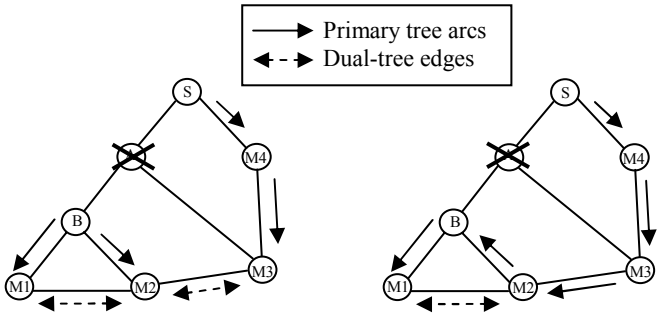
1. SA_B ← {M1, M2}, PA_B ← {M3}, NA_B ← {M4}
2. bp_B ← (M2, M3, M4)
   { (M2,M3, M4) is the shortest path which interconnects one node in SA_B to one node in NA_B }
3. do nothing { because bp_B != *infinite path* }
4. bp_B ← (M2, M3)
   { because bp_B includes node *M3* which belongs to PA_B }
5. e_1 ← M2, e_2 ← M3
6. Reconfig message is created. The list of nodes included in the message is: [B, M2, M3]
7. Reconfig message is sent to node M2 which is the successor of B in the Reconfig message list.
8. parent(B, T_p) ← M2
   { node *B* sets *M2* as its parent in T_p }
9. del_child(B, T_p, M2)
   { *M2* is removed from the child list of *B* in T_p }

When node *M2* receives the Reconfig message sent by node *B*, it runs the routine depicted in Algorithm 2.

M2 execution trace of Algorithm 2

1. Reconfig message is sent to node M3 which is the successor of M2 in the Reconfig message list
   parent(M2, T_p) ← M3
   { node *M2* sets *M3* as its parent in T_p }
   del_child(M2, T_p, M3)
   { do nothing because *M3* is not a child of *M2* in T_p }
2. add_child(M2, T_p, B) { *B* becomes a child of *M3* in T_p }

As in step 1 of Algorithm 2, *M2* forwards the Reconfig message to *M3*, this latter runs also Algorithm 2:



(a) Situation at the end of M3 recovery procedure and before the detection of the failure by B

(b) Situation at the end of the recovery procedure

Fig. 3. Improved dual-forest protection

M3 execution trace of Algorithm 2

1. do nothing { because $M3 = e_2$ }
2. add_child(M3, T_p, M2)
   { M2 becomes a child of M3 in T_p }

At this point of execution, the recovery procedure ends with success. Indeed, the configuration results illustrated in Fig. 3(b) show that the source node can deliver multicast packets to all multicast group members.

B. An efficient heuristic for dual-forest computation

The restoration algorithm of the improved dual-forest uses paths interconnecting primary leafs for recovery. These paths which form the backup structure are built with only the use of nodes and links of the reduced topology. Thus and in order to maximize the level of protection with the improved dual-forest protection scheme, the number of connected components of the backup structure must be equal to the number of connected components of the reduced topology.

In this section, we endeavor to determine an efficient heuristics to pre-compute the backup structure which do not decrease the protection level (the backup structure must provide the same protection level as that of the reduced topology). Thus, these heuristics have the objective of optimizing the two following criteria under the constraint of maximizing protection: the cost of the backup structure (criterion 1) and the cost increase of the multicast tree after recovery (criterion 2).

To optimize the first criterion under the constraint, the backup structure must correspond to a *Steiner forest*. We recall that a Steiner forest is a forest of Steiner trees, one tree per connected component.

To optimize the second criterion without decreasing the level of protection, the backup structure must include for each possible failure, at least one of backup path sets which minimizes the cost increase of the multicast tree after recovery. In the improved dual-forest protection case, the multicast tree cost after recovery is equal to the residual primary tree cost plus the cost of path(s) bp_{xi} deduced by the restoration algorithm of the improved dual-forest (Algorithm 1 of section III.A). Since the residual primary tree cost is given and depends only on the failure, the optimization of the

multicast tree cost increase involves the cost minimization of the path(s) bp_{xi} chosen by Algorithm 1 of section III.A.

There exist different backup structures belonging to the reduced topology which include a set of backup paths minimizing the multicast tree cost increase after recovery. For instance, the backup structure corresponding to the reduced topology graph is a trivial solution which is not interesting since its cost is very high. Here, we search for a backup structure (BS_{opt}) which optimizes the above two criteria without decreasing the level of protection. Thus, the heuristic may determine an approached Steiner forest (criterion1) which includes backup paths minimizing the cost increase of the multicast tree after recovery (criterion 2).

To determine BS_{opt}, we distinguish the link failure case from the node failure case. Whereas in the link failure case it must be made sure that a path of minimal cost allowing the recovery belongs to BS_{opt}, it is necessary to be sure in the case of node failure that a Steiner tree (a set of paths) allowing the recovery belongs to BS_{opt}.

1) Link failure case

In case of link failure detected by node x, the backup path minimizing the multicast tree cost increase and allowing the recovery is the shortest path belonging to the reduced topology which interconnects one node in SA_x to one node in ($NA_x \cup PA_x$); therefore, such path (or an equivalent path in cost) must belong to BS_{opt}. As x can be any node (different from the source) of the primary tree, we conclude that BS_{opt} must include for each value of x, one shortest path interconnecting one node in SA_x to one node in ($NA_x \cup PA_x$). The property below ensures that BS_{opt} in the case of link failure corresponds to the minimal KMB forest which covers all primary leafs in the reduced topology.

Property:

Let G an undirected graph and let T be a tree built on G. Let {E1, E2} be a partition of the set of T leafs. The cost of the shortest path interconnecting one node in *E1* to one node in *E2* in the reduced graph (obtained by the elimination of all links and inner nodes of T) is equal to the cost of the shortest path interconnecting one node in *E1* to one node in *E2* in the KMB forest which covers all T leafs.

Proof: (by construction)

Suppose that the cost of the shortest path interconnecting one node of *E1* to one node of *E2* in the reduced graph is c and suppose that all paths of cost lower than c (which do not create loops) are already added to the building KMB forest. At this moment, we know that no node of *E1* is connected to any node of *E2*. In the next steps of the KMB forest building, a path of cost c will be chosen to interconnect one node of *E1* to one node of *E2* since the nodes of the 2 sets are completely disjoint (no risk of loop formation).

It is trivial that if there is no path interconnecting the two sets in the reduced graph, there will be no path interconnecting them in the KMB forest. ■

Since $\{SA_x, NA_x \cup PA_x\}$ forms a partition of T_p leafs independently of values of x, we deduce from the property above that the cost of the shortest backup path allowing recovery in the reduced topology is equivalent to the cost of the shortest backup path allowing recovery in the KMB forest. Hence, if the backup structure used for protection corresponds to the KMB forest which is an approached Steiner forest (criterion 1), one backup path minimizing the multicast cost increase will be a candidate for recovery (criterion 2). This path is often selected but there are cases where we prefer another path in order to ensure recovery.

2) Node failure case

In the case of node failure, the KMB algorithm is also an efficient heuristic to compute the dual-forest. Due to the lack of space, we give here only the main ideas which are behind this assertion. Further details can be found in [9].

When a node failure is detected by a downstream node x of the primary tree, the set of backup structures which first minimize the cost increase of the multicast tree after recovery and second repair the multicast communication is the set of minimal forests $\{MF_x^i\}_{i>0}$. Each forest of $\{MF_x^i\}_{i>0}$ is composed of Steiner trees so that all nodes of ($SA_x \cup PA_x$) are covered by the forest and exactly one node of NA_x belongs to each Steiner tree (of the forest).

As a result, for each inner node x all links of one forest in $\{MF_x^i\}_{i>0}$ must belong to BS_{opt} which is a Steiner forest. Thus and since a sub-tree of a Steiner tree is not necessarily a Steiner tree, we conclude that, in general, there is not an exact solution which optimizes the two criteria together. However, among Steiner tree heuristics there is one (KMB heuristic [8]) which can be used to optimize the two criteria. Indeed, any sub-tree of a KMB tree which interconnects a sub-set of Steiner nodes of a KMB tree is also a KMB tree (its Steiner nodes belong to the sub-set).

IV. Simulation

In order to evaluate the quality of the improved dual-forest protection (IDFP), we choose to compare it with the two techniques presented in section II.

A. Comparison criteria

The following two metrics have been selected in order to evaluate the quality criteria of the IDFP, DTP and PP schemes: protection rate (PR) and average tree cost increase after restoration (ATCI).

PR measures the survivability of the network, the higher is this rate the better is the protection. It is defined as a ratio between the number of cases where the protection technique successes to repair the multicast tree and the total number of (failure) cases.

The tree cost increase is determined as the ratio between the cost of routing structure used after the restoration and the cost of tree used before the failure detection. A significant ATCI indicates a wasting of bandwidth.

B. Simulation model

In the simulation presented here, all links are bi-directional and of cost equal to 1 (without lack of generality, our model

can be easily extended to any link cost). 400 connected graphs are randomly generated with the Waxman approach [10] in which a link between two nodes x and y is chosen to be in the graph according to probability $p(x, y) = \beta.exp(-d(x, y) / \alpha L)$, where $d(x, y)$ is the Euclidean distance between x and y, L is the maximum distance between any two nodes, α and β are parameters verifying $0 < \alpha, \beta < 1$. α is chosen equal to 0.25 to have connectivity characteristics of Internet networks and we varied β and the size of graphs to have different average node degrees.

After that, we vary the multicast group size between 2 (sparse mode) and 30 (dense mode), and we build randomly for each graph and each size 100 multicast groups. One node of the multicast group is randomly chosen to be the source. For each multicast group (and each graph), the primary tree is computed using the Dijkstra shortest path tree algorithm.

Two types of failures are considered: link failures and node failures. The failing links are selected randomly among the primary tree links and the failing nodes are chosen randomly among the inner nodes of the primary tree (nodes different from the source and leaf nodes). For each failure type, the different protection schemes are used for recovery and the metrics described in sub-section IV.A are computed.

As the conclusions concerning the best protection scheme are the same in all our simulation cases (graph size equal to 50 and 100, and average node degree equal to 2.75, 3, 3.5, 4, 4.4 and 8), we present here only the results depicted in Fig. 5, Fig. 6 and Fig. 7 obtained for β equal to 0.08 and graph size equal to 100 (the average node degree is equal to 4.4).

C. Comparison and analysis

1) Dual-tree protection Vs Dual-forest protection

The results depicted in Fig. 5 show a great difference between the PR of IDFP and that of DTP in the two types of failures (link and node failures). Indeed and except in the unicast case (group size equal to 2) where the two schemes act in the same way for restoration, the performances of IDFP are widely better than those of DTP.

Note that the usage of DTP is drastically restricted because of the fast decrease of its PR. In Fig. 5 for instance, the group size must be lower than 5 to have an average protection rate upper than 0.75. The reason of the fast decrease of the PR in DTP is due essentially to the augmentation of the primary tree size with the increase of the multicast group size. Indeed, the probability to obtain a connected reduced graph (and thus a dual-tree) after the elimination of all links and inner nodes of a primary tree is smaller when the primary tree is larger.

Since the PR of the DTP is small and undesirable, we focus in the rest of simulation only on the comparison between the PP and IDFP schemes.

2) Path protection Vs Dual-forest protection

Concerning the protection rate metric (Fig. 6), the IDFP is always better than the PP.

The values of PR are identical for a group size equal to 2 in both link and node failure types because the IDFP and the PP act in the same manner to restore unicast communication.

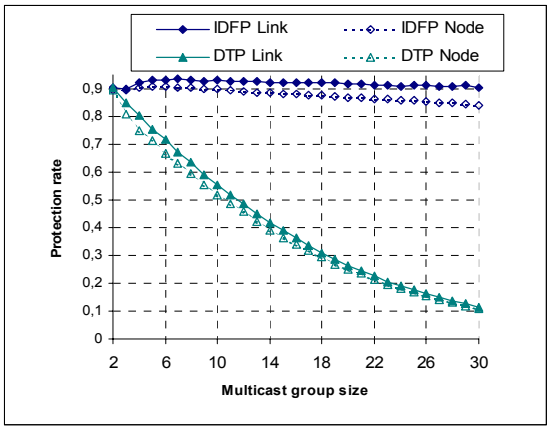


Fig. 5. Protection Rate (PR)

When the size of multicast groups is higher than 2, the curves of the IDFP go up rapidly until the group size reaches value 5. This can be explained by the augmentation of the number of primary leafs as the group size increases. Adding a new leaf node can then make possible for a non protected primary leaf node to have a backup path by its interconnection to this new primary leaf node.

When the group size is higher than 5, the protection rate of the IDFP seems to be stabilized with very light diminution especially in the case of node failure. That is due to the primary tree size which increases and which becomes significant, restricting the reduced topology on which the dual-forest is built. The PR in the case of failure node decreases more rapidly than in the case of link failure because a node failure involves in general the reconfiguration of more than one backup path whereas only one backup path is used to bypass a link failure.

Contrarily to IDFP curves, the PP ones go down continuously. Indeed, as each member node is protected by a distinct path, the probability of success of the restoration procedure (probability to determine a backup path for each affected member) decreases with the increase of the size of affected members which depends on the multicast group size.

The last important point is related to the difference between IDFP curves and ideal ones which does not exceed 0.06 in the link failure case and 0.1 in the node failure case. We recall that an ideal protection technique will ensure the restoration of the multicast communication after a failure if all members belong to the same connected component in the topology graph obtained after the failure.

With regard to the multicast tree cost increase (Fig. 7), the curves of the two techniques (IDFP and PP) have a similar form with a significant and almost constant difference (≈ 0.1).

The IDFP introduces small increases in the ATCI for groups of small size. This is due to the distance between the primary leafs which is more important for groups of small size. This increase in tree cost is not awkward because the cost of primary tree is small (ATCI is a ratio). For groups of medium and large sizes, the ATCI is negligible and in most

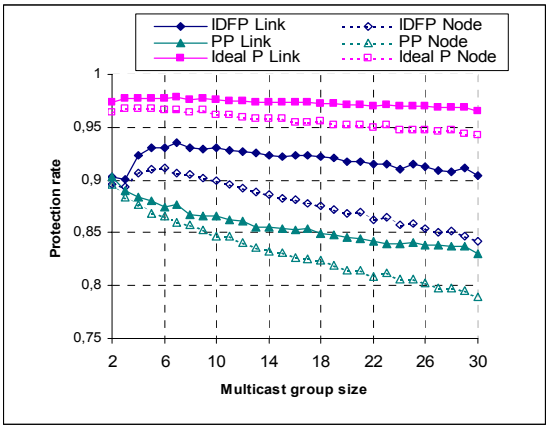


Fig. 6. Protection Rate (PR)

cases of node failure lower than 1. This is due to the close distance between primary leafs in one side, and to the use of a KMB forest as backup on the other side. Indeed, the replacement of an affected part of the multicast tree which is computed according to Dijkstra algorithm by another part determined with the use of a KMB forest decreases in most cases the cost of the multicast tree.

However, in PP the backup paths have a higher cost than the primary ones since they are computed using a restricted topology (while the global topology is used for the computing of primary paths). As a result, a significant tree cost increase is observed in PP (more than 11% of the total quantity of bandwidth allocated on the multicast tree is needed for a recovery from only one failure).

V. Conclusion

In this article, we gave an overview of the existing pro-active protection schemes for multicast. We outlined the problems and insufficiencies of each scheme and we proposed the improved dual-forest for multicast protection which makes several improvements to the traditional dual-tree protection in order to solve most of its problems. Indeed, our protection scheme can cope rapidly with both node and link failures without loop formation. The level of protection is also increased with the use of a forest instead of a tree as a backup

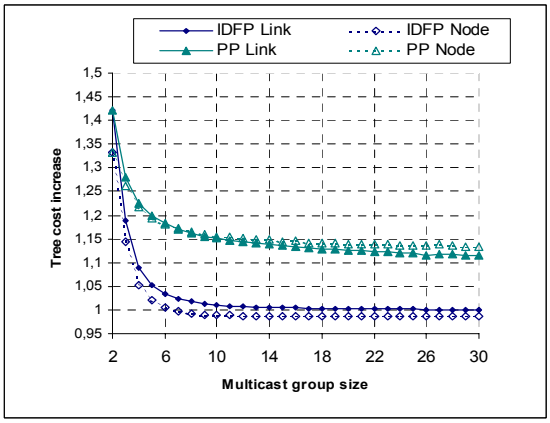


Fig. 7. Average Tree Cost Increase (ATCI)

structure. Moreover, we have elaborated that the use of KMB heuristic to compute the dual-forest is a good technique which decreases both the cost of the backup structure and the cost of the multicast tree after restoration.

We evaluated the performances of our protection scheme by comparing it to the dual-tree and to the path protection schemes. The results show that the improved dual-forest protection is a promising protection scheme since it has a better protection rate than those of the dual-tree and path protection schemes. Its tree cost increase after recovery is also better than that of path protection.

An improvement can be made to the dual-forest protection in order to enhance its protection rate. Typically, the contraction of a (primary)sub-trees already protected into a contracted leaf nodes can provide a better protection rate.

VI. References

[1] P. Meyer, S. Van Den Bosch, and N. Degrande. High Availability in MPLS-Based Networks. Alcatel Telecommunication Review, 4th Quarter 2004.

[2] K. Murakami and H. S. Kim. Optimal capacity and flow assignment for self-healing ATM networks based on line and end-to-end restoration. In IEEE/ACM Trans. on Networking, vol.6 (2), pp.207-221, April 1998.

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

[4] C. Wu, W. Lee, Y. Hou, and W. Chu. A New Preplanned Self-healing Scheme for Multicast ATM Network. In Proceedings of IEEE ICCT'96, vol.2, pp.888-891, May 1996.

[5] C. Wu, W. Lee, and Y. Hou. Backup VP Preplanning Strategies for Survivable Multicast ATM Networks. In Proceedings of IEEE ICC'97, vol.1, pp.267-271, June 1997.

[6] A. Fei, J. Cui, M. Gerla, and D. Cavendish. A "Dual-Tree" Scheme for Fault-Tolerant Multicast. In Proceedings of IEEE ICC 2001, June 2001.

[7] M. Medard, S. Finn, R. Barry, and R. Gallager. Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. IEEE/ACM Transactions on Networking, vol.7(5), pages 641–652, October 1999.

[8] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for Steiner trees. Acta Informatica, vol.15, pages 141-145, 1981.

[9] M Y. Saidi, B. Cousin, and M. Molnár. An Efficient Multicast Protection Scheme based on a Dual-Forest. IRISA Internal Research Report N° 1786, March 2006.

[10] B M. Waxman, Routing of multipoint connections, IEEE JSAC, vol.6 (9), Pages 1617-1622, December 1988.