Efficient protection of many-to-one communications

Miklós MOLNÁR, Alexandre GUITTON, Bernard COUSIN, and Raymond MARIE

Irisa, Campus de Beaulieu, 35 042 Rennes Cedex, France

Abstract. The dependability of a network is its ability to cope with failures, *i.e.*, to maintain established connections even in case of failures. IP routing protocols (such as OSPF and RIP) do not fit the dependability objectives of today applications. Moreover, forwarding techniques based on destination address (like IP) induce many-to-one connections. If a dependable connection is needed, all primary paths and protections having the same destination must be established in a coordinated way. In this paper, we propose a fault recovery for many-to-one connections based on a cold (preplanned) protection. The main advantage of our approach is that the recovery in case of failures is achieved within a short delay. Additionally, with respect to other approaches, the dependability of the routing scheme is increased in the way that it statistically copes with many failures. The algorithm we propose computes an efficient backup for an arbitrary primary tree using an improved multi-tree algorithm.

Keywords: network, fault-tolerant routing, many-to-one, cold protection, multi-tree algorithm

1 Introduction

High-speed networks are becoming increasingly important and allows the development of applications with real-time constraints, such as multimedia services, cooperative systems, distributed computing. These applications often rely on the survivability of the network: communications should not be interrupted for a long time by a failure of a link or of a router. Indeed, the longer the communication is interrupted, the more packets are dropped. The problem of fast recovery has been well studied for several types of communications, including broadcast (one-to-all), unicast (one-to-one) and multicast (one-to-many). However, there is no efficient proposition for dependable incast (many-to-one) communications.

Incast connections are many-to-one, *i.e.*, several sources send data to a single destination. An incast connection can be the support of homogeneous or heterogeneous communications. Examples of applications inducing homogeneous communications include log collection, data gathering in sensor networks, auction sales and massive submissions. In networks where the forwarding of packets is based on their destination address (such as IP networks), all the communications toward the same destination form an incast connection. In this case, the connection is heterogeneous since it is composed of communications having different requirements and protocols. For example, a FTP communication from an host A to an host C and a HTTP communication from an host B to an host C form an incast connection. Incast connections are traditionally realized using a tree[1].

Implementing dependable communications is a major thread for current networks. Indeed, the network is supposed to be survivable, *i.e.*, it should withstand failures of links or routers. Two measures of the dependability of a network can be considered: the recovery delay and the number of failures managed [1]. It is therefore critical to reduce the recovery delay as much as possible. Classical recovery delays of IP protocols such as OSPF or RIP reach tens of seconds (see [2] for OSPF and the slow convergence problem [3] for RIP). The other measure, the number of failures managed by a recovery mechanism, impacts on the reliability of the network. In our model, we consider two types of failures: independent failures and highly correlated failures. There is a trade-off for the recovery mechanisms in protecting efficiently against independent failures and highly correlated failures.

Our objective is to recover quickly from failures on an incast communication while coping with as much failures as possible. In this paper, we propose a cold preplanned protection that allows local recovery using arc-disjoint trees. Our construction of arc-disjoint trees is a generalization of the algorithm described in [4].

Section 2 gives a state of the art on dependable communications. Section 3 describes our protection construction based on arc-disjoint trees.

[1] Multicast connections are also realized using a tree, but incast connections and multicast connections are not symmetric: multicast connections and multicast routing protocols require a particular mechanism in routers, the duplication mechanism, while incast connections do not require any.

Section 4 describes our local recovery mechanism. Section 5 analyzes the trade-off of protecting independent failures and highly correlated failures. Finally, we conclude our work in Section 6.

2 State of the art of dependable connections

Several ways to cope with failures exist. In this section, we briefly survey the existing approaches to realize dependable connections while concentrating on our main concern: the fastness of the recovery. A detailed survey on survivability can be found in [5] (in the case of WDM networks).

Hot and cold redundancy. Hot redundancy, denoted 1+1 redundancy, consists in sending each message on two disjoint paths simultaneously (*cf.* [6] for an example). Hot redundancy allows fast recovery since the destination receives the packet from one of the paths even if a failure on the other path has occurred and has not yet been detected. However, hot redundancy wastes a lot of bandwidth. This major drawback and the necessity of a selecting algorithm at the destination make hot redundancy not very used.

Cold redundancy, denoted 1:1 redundancy, consists in raising a recovery mechanism once a failure is detected (*cf.* [6] again for an example). Although 1:1 redundancies are slower than 1+1 redundancies because of the failure detection delay. They are often preferred since they save bandwidth. The restoration and protection are the two main types of cold redundancy.

Restoration and protection. Restoration is a reactive approach to cope with failures. At the time a failure is detected, the router that detected the failure searches for a new path to reroute the traffic to the destination [7]. The advantage of restoration is that it adapts to the current state of the network. However, intensive computations are required for the router to find a new path, which increases the recovery delay. Usual Internet routing protocols use this approach.

Protection is a proactive approach to cope with failures. The behavior of the routers in case of a failure is preplanned [8]. At the time the failure is detected, the router reroutes the traffic to the preplanned protection path. This approach has the advantage of being very fast, since the recovery is raised without any additional computation of the router [9]. However, less failures can be managed compared to restoration since protection is proactive. Classic protections are end-to-end or local.

End-to-end and local. End-to-end recovery consists in rerouting the traffic at the source on an arc-disjoint path, once a failure is detected. A typical example of end-to-end protection is the path-based protection. The delay induced by the end-to-end recovery is high because the source has first to be informed that a failure occurred on the primary path before raising the recovery. Another drawback of path-based protection is that it cannot cope with two successive failures: if a failure occurs on the primary path and another occurs on the alternate path, the connection is interrupted.

Local recovery consists in rerouting the traffic at the router that detected a failure. A typical example of local recovery is the link-based protection. The delay induced by the local recovery is low because the recovery is raised locally. Using link-based recovery, several successive failures on the primary path can be managed, as long as they concern different links. A drawback of link-based protection (but not of local recovery approach) is that node failures are not managed. A comparison of link-based protection and path-based protection can be found in [10].

In this paper, we propose a new local cold protection to recover from failures on incast connections within a short delay, and without the drawback of the link-based protection. The specificity of incast connections is the large number of sources; therefore, traditional end-to-end recovery mechanisms are not suited to incast, where all the sources have to be informed of the failures that occurred.

3 Proposed protection of incast trees

Our proposition aims the construction of dependable incast connections. We show later that the dependability of the routing scheme is reinforced since the proposed protection statistically withstands many failures.

Many-to-one connections require the establishment of the primary paths in a coordinated way. The backup paths should be synchronized together and also with the primary paths. We call primary tree the set of primary paths.

For basic incast connections, the primary tree is usually a shortest path tree. QoS aware incast connections may use different partial spanning trees, depending on the network status. For this reason, we assume that the primary incast tree is given to our algorithm either by the application or by the network management. Often, this primary tree spans only a sub-graph of the given network. The objectives of the protection are: (i) it should work for any topology and (ii) it should work for any given primary tree (partial or not).

In this section we present how our protection can achieve these objectives. Since it refers to the multi-tree construction algorithm proposed in [4], a brief description of the algorithm is done in the following.

3.1 Basic multi-tree construction

A multi-tree is a set of two directed trees that are arc-disjoints. The algorithm presented in [4] describes a way to compute these two trees. The computation works only in edge-redundant topologies and the two trees spans all the nodes of the network. The algorithm assumes that all links are bidirectional.

The multi-tree is built by adding successively external paths, as specified by Algorithm 1. An external path is a path starting in a spanned node u, ending in a spanned node v and such as all intermediate nodes are not yet spanned by the multi-tree. Generally, $u \neq v$ (except at the first iteration or in the case of articulation vertex where $u = v = r$, r being either the root or the articulation node). From each external path, two arc-disjoint branches are extracted such as one of them ends at u while the other ends at v.

Algorithm 1 Multi-tree construction.

```
initialize the multi-tree with the destination node
while an external path of the current multi-tree exists do
  select an external path $p$
  extract two arc-disjoint directed paths from $p$
  add the two directed paths to the multi-tree
end while
```

Figure 1 shows the successive steps of the algorithm. In this example, an external path is chosen arbitrarily at each step. The corresponding two arc-disjoint branches are shown on the figure. One of the tree of the multi-tree is represented in solid lines (and is referred to as the blue tree in [4]) while the other is represented in dashed lines (and is referred to as the red tree in [4]).

Advantages. With the help of the multi-tree, tree-based communications can be protected against node and link failures in edge-redundant graphs, and the multi-tree is relatively easy to compute. It can be applied to realize hot-redundancy for broadcast or incast communications or as a preplanned protection for cold-redundancy.

Drawbacks. One of the drawbacks of the described multi-tree protection is that the algorithm does not deal with arbitrary topologies, only with edge-redundant topologies.

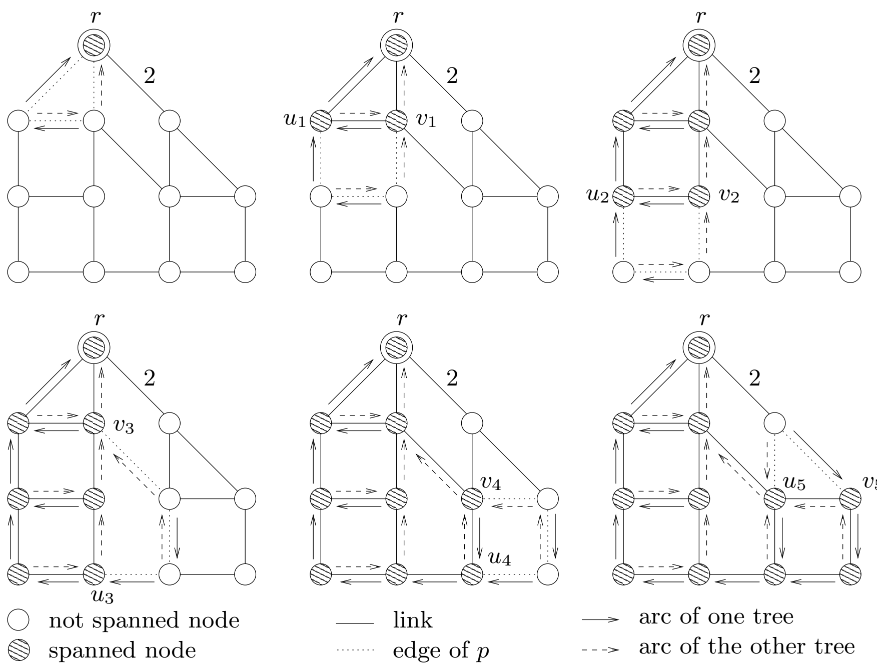

Fig. 1. The multi-tree construction.

A more important drawback follows from the fact that the selection of successive external paths is not determined. Thus, the diameter of the trees (and as a consequence, the length of the primary paths) can be arbitrary large (for example, it can be seen on Figure 1 that the distance in the solid tree from node v_5 to the destination r is very long). The QoS requirement of the applications or the network management often impose the use of particular primary trees (e.g., of shortest path trees or QoS aware trees).

Finally, in most incast communications, only a sub-set of nodes belongs to the set of the sources and so the incast tree is a partial spanning tree. The algorithm should be adapted to partial spanning trees.

In the following, we will specify how the different drawbacks of the multi-tree based protection can be eliminated.

3.2 Extension to arbitrary topologies

Often, in real network, articulation nodes or edges are possible. Even in initially redundant networks, articulations can be produced due to persistent failures. The protection computation should work also in these cases.

To build a multi-tree in an arbitrary connected network, we propose to add to the previous algorithm a particular case when articulation edges are found. In this case, we propose the creation of two directed arcs on the articulation edge toward the destination. This edge is not protected against failure but no algorithm could have protected it. However, if failures occur on other parts of the network, they can be recovered. It can be noticed that the two trees of the multi-tree are not arc-disjoints anymore.

3.3 Protection of a given incast spanning tree

In network, failures are rare. Therefore, the chosen tree used if no failure occurs has to ensure an efficient delivery of data packets and certain QoS criterion. Generally, none of the directed trees built by the multi-tree algorithm does correspond to a good primary tree and can not be used for the communications. However, computing a multi-tree can help in finding the backup support of the desired primary tree as presented in the following[2].

To adapt the protection to a given (total) primary tree (for example to the shortest-path tree), we propose the construction of a multi-tree spanning all of the arcs of the given primary tree. Our dependable incast connection computation contains two steps: (i) the computation of a multi-tree spanning a given primary tree and (ii) the construction of a backup forest on the basis of the multi-tree.

Multi-tree construction for a given primary tree To ensure that the primary tree T_p is covered by the multi-tree, we have to ensure that T_p is covered by the union of the external paths selected during the multi-tree construction. In other words, we have to ensure that every arc of T_p is covered by an arc of the multi-tree.

Let us denote by $dist(T_p, r, n)$ the hop distance from r to a node n in T_p. At each iteration, the algorithm selects an external path containing exactly one edge (n_1, n_2) that is not in the primary tree T_p and with at least one node of $\{n_1, n_2\}$ not spanned by the multi-tree. One can prove that this kind of external path exists if the topology is redundant and the primary tree spans it. If there are several candidates, then the external path minimizing $dist(T_p, r, n_1) + dist(T_p, n_2, r)$ is selected. Figure 2 shows such an edge on an example where the primary tree T_p corresponds to the

[2] To simplify let us suppose here that, similarly to the multi-tree computation, all nodes of the network will participate to the incast communication. The case of partial spanning trees is discussed in the next sub-section.

shortest-path tree. Having chosen the not yet spanned edge (n_1, n_2), the external path p from u to v can be found as follows: u and v are spanned by the multi-tree, p contains the path in the primary tree from n_1 to u, the edge (n_1, n_2) and the path from n_2 to v.

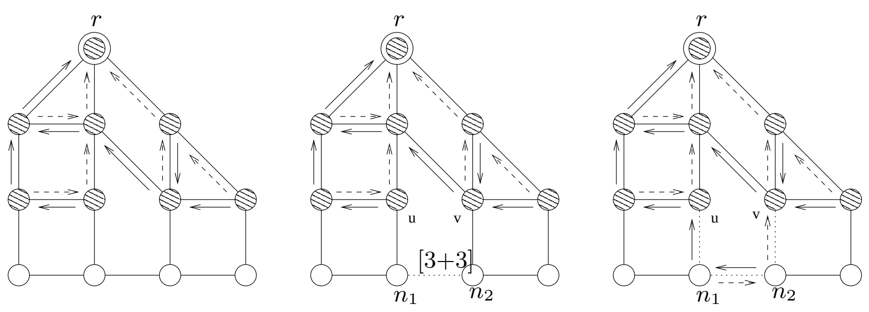

Fig. 2. An external path on a shortest-path tree.

Minimizing the hop distance $dist(T_p, r, n_1) + dist(T_p, n_2, r)$ on T_p allows the protection to be dense. A dense protection is more robust in the case of multiple failures than a sparse protection.

Property 1 shows that our selection of external paths ensure that the primary tree T_p is covered by the multi-tree.

Property 1. The arcs of the directed primary tree T_p are in the multi-tree built by our algorithm.

Proof. There is only one outgoing arc for each node of the tree T_p directed to the root. The computed multi-tree spans all the nodes by successive external paths. To prove the property, it is sufficient to show that adding a node to the multi-tree implicates also adding its outgoing arc in the primary tree T_p. Let p be an external path from the node u to v selected at any iteration of the algorithm. In a first time, we show that the outgoing arc of any node $n \in p$ is in p and in a second time we show that there is an arc directed to the same direction in the multi-tree.

- Let us assume that the outgoing arc of n in T_p is not in p. In this case, the adjacent edge of $n \in p$ are not in T_p but this is impossible because there is only one edge in p which is not in T_p.
- The multi-tree algorithm associates two outgoing arcs (one in each directed tree of the multi-tree) to each node, except u and v. The outgoing arc on T_p of any node different from u and v corresponds to an arc in the created multi-tree.

Construction of the backup forest Our improved multi-tree algorithm builds a multi-tree $M = (T_1, T_2)$ covering the primary tree T_p. To obtain the backup support of the primary tree, Algorithm 2 is proposed:

Algorithm 2 Backup forest construction.

```
Require: $T_p$ a primary tree, $M = (T_1, T_2)$ a multi-tree covering $T_p$
  $F \leftarrow arcs(T_1) \cup arcs(T_2)$
  $F \leftarrow F \backslash arcs(T_p)$
```

The first step of the algorithm merges the arcs of T_1 and T_2 into the directed set F. T_1 and T_2 are disjoint, except from the arcs on the articulation edges. Then, in F, only one arc exists on the articulation edges. The second step of the algorithm removes in F the primary arcs of T_p. The remaining arcs are the backup protection of T_p. It can be noticed that F is a forest.

3.4 Partial spanning trees

Generally, only a sub-set of the nodes participates to the incast communication. To create a partial multi-tree which covers at least the partial primary tree, we propose two algorithms.

A simple solution can be obtained by computing the multi-tree spanning all the nodes and by pruning the parts that are not used or redundant. This solution requires an important computation even if there are few nodes in the set of sources[3].

A more efficient computation can be obtained if only the required part of the multi-tree is built. The partial multi-tree can be built with the help of successive shortest loops from the destination to the sources following the primary paths in the reverse direction. The external path selection and the stop condition of the multi-tree algorithm should be modified accordingly. Nodes which are in the partial multi-tree but not in the primary tree have two directed paths to the destination. The outgoing arcs which is not in the primary tree can be deleted because the protection of these nodes is not needed. Other outgoing arcs are used as backup segments for the primary tree[4]. Figure 3 illustrates the partial multi-tree construction and the obtained protection scheme. The left part of the figure shows the partial multi-tree construction. The primary tree T_p, in thick lines, is contained in $M = (T_1, T_2)$. The right part of the figure, the

[3] In this case, proof of Property 1 remains the same, as long as the pruning does not alter neither the primary tree nor the backup forest.

[4] In this case, the proof of Property 1 is not valid anymore. However, a similar proof can be done by noticing that the last part of the external path corresponds to the path in the primary tree T_p.

obtained protection scheme, shows the primary tree T_p in solid lines and its backup F in dashed lines.

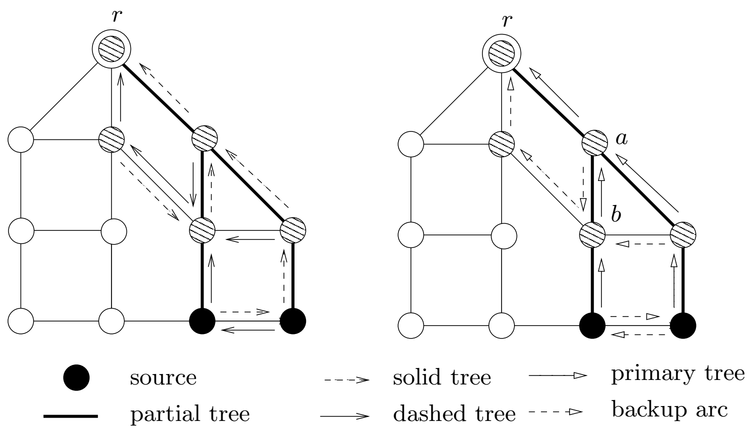

Fig. 3. Multi-tree and backup construction for a partial incast tree.

4 Proposed recovery mechanism

The previous section described how to compute a protection for a given tree. Once the protection has been configured, a local recovery mechanism has to be implemented in routers to protect the connection against failures. Let us recall that the protection ensures that, for every node n of the primary tree, there exists two arc-disjoint paths from n to the destination r of the incast connection.

We assume that all the routers can store two entries for every destination. The primary entry for the node n corresponds to the first edge on the primary path toward the destination and the alternate entry corresponds to the first edge of the backup path. If a router detects a failure on the primary entry for a destination r, it switches its primary entry to its alternate entry.

A router detects a failure propagation need when it receives a packet for a destination r on an interface which corresponds to the next hop for r on the primary path. In this case, it switches its primary entry to its alternate entry to avoid loops. This failure propagation occurs only when the backup path uses the edges of the primary path in the reverse direction. For example, if node a of Figure 3 detects that the link (a, r) failed, it switches to its alternate entry (failure detection). Then, the packets that reach a are forwarded to b. When receiving a packet from a for r, b detects that a is the next hop to r on its primary path. Then, b switches to its alternate entry (failure propagation). The packets follow the backup path up to r.

Now that we described our protection and our recovery, the next section analyzes its behavior in the case of independent or highly correlated failures.

5 Analysis of the dependability in the case of multiple failures

In this section, we analyze the dependability of several protections in case of multiple failures. The number of failures that can be managed by a protection greatly depends on the network topology and on the incast connection. To study the dependability of a protection in the case of multiple failures, we propose to discuss on the number of failures that do not interrupt communications.

5.1 Independent failures

In our model of independent failures, we assume that failures occur successively on the primary path. Indeed, failures that do not occur on a primary path do not impact communications.

Path-based protection When the first failure occurs on the primary path and the source is informed, the traffic is swapped to the backup path at the source. Then, if an independent failure touches the backup path, the communication is interrupted.

Link-based protection Link-based protections do not cope with node failures. In the case of a link failure, the bypass is used to reach the next node while the rest of the primary path is used. If an ulterior link failure occurs on the primary path, another bypass can be used if and even if the second bypass is link-disjoint with the first bypass. The link-based protection allows independent link failures if the bypasses are independent. However, it does not cope with failures which occurs on the bypasses.

Our protection The proposed incast tree protection copes with a link or a node failure. If a second failure touches the backup used to recover from a first failure, there are two possibilities. If this backup corresponds to a primary path of another communication and its backup is different from the first primary path, then the local recovery is possible. If the failed part of the backup does not correspond to a primary path of another communication or if the backup of this failed point uses the first failed primary path, then the mechanism cannot cope with the failure.

The mechanism is illustrated on Figure 4 (A) and (B) in the case of link and node failures.

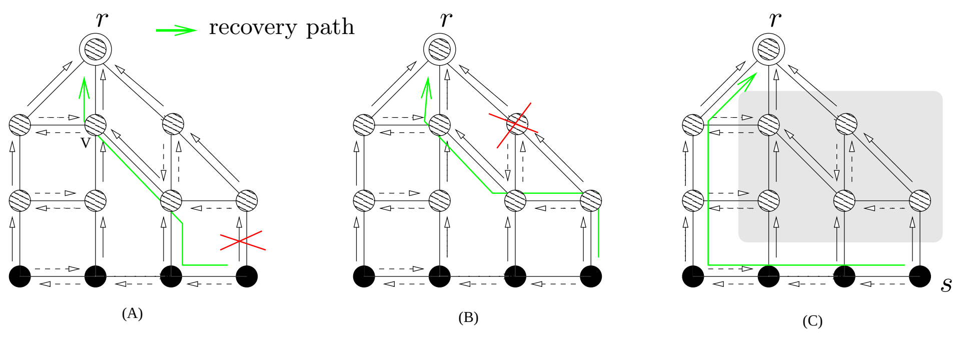

Fig. 4. Recovery capability of our mechanism.

5.2 Highly correlated failures

In our model of highly correlated failures, the failures occur simultaneously on adjacent links or nodes of the network: a connected subgraph of the network fails. The higher the protection resists to highly correlated failures, the higher can be the diameter of the failed subgraph without interrupting communications. In the same way than previously, we assume that at least a link or a node of primary tree fails.

Path-based protection Simultaneous failures can occur on the links and nodes of a primary path. The path based protection copes with large diameter of failures when all internal nodes and edges of the primary path failed. However, the alternate path should be intact.

Link-based protection Since link-based protection does not cope with node failures, the maximal diameter of a recovered failure is one link.

Our protection In redundant topologies, the proposed tree-based protection gives a link-disjoint alternate path from all the nodes of the primary path to the destination. Highly correlated failures on a primary path with large diameter can be recovered. Moreover in the case of dense incast communications, the mechanism copes with failures which occur on the alternate paths. Figure 4 (C) shows the maximal sub-set of the network which can fail with recovery from a source s.

6 Conclusion

In this paper, we presented a cold protection for many-to-one communications. The protection computation is based on the construction of a backup forest, given an arbitrary primary tree. The recovery uses a simple failure propagation mechanism and can be realized by a local switch operation in the concerned routers. Thus, it produces low failure recovery delay. We studied the impact on the protection of two scenarios of failures: independent failures and highly correlated failures. We were able to show that our method can propose a trade-off between managing independent failures and highly correlated failures. In the future, we intend to evaluate quantitatively our protection through simulations.

References

1. Duato, J.: A theory of fault-tolerant routing in wormhole networks. IEEE Transactions on Parallel and Distributed Systems **8** (1997) 790–802
2. Goyal, M., Ramakrishnan, K.K., Feng, W.C.: Achieving Faster Failure Detection in OSPF Networks. In: IEEE ICC. (2003)
3. Obradovic, D.: Formal Analysis of Convergence of Routing Protocols. PhD thesis, University of Pennsylvania (2001)
4. Médard, M., Finn, S.G., Barry, R.A.: Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. In: IEEE/ACM Transactions on Networking. Volume 7. (1999) 641–652
5. Maier, G., Pattavina, A., De Patre, S., Martinelli, M.: Optical Network Survivability: Protection Techniques in the WDM Layer. Photonic Network Communications **3/4** (2002) 251–269
6. Anand, V., Qiao, C.: Static versus dynamic establishment paths in WDM networks, Part I. In: IEEE ICC. Volume 1. (2000) 198–204
7. Ramamurthy, S., Mukherjee, B.: Survivable WDM Mesh Networks, Part II – Restoration. In: IEEE ICC. Volume 3. (1999) 2023–2030
8. Ramamurthy, S., Mukherjee, B.: Survivable WDM Mesh Networks, Part I – Protection. In: IEEE INFOCOM. Volume 2. (1999) 744–751
9. Nakamura, R., Ono, H., Nishikawara, K.: Reliable switching services. In: IEEE Globecom. (1994)
10. Johnson, D., Johnson, G.N., Beggs, S.L., Botahm, C., Hawker, I., Chng, R.S.K., Sinclair, M.C., O'Mahony, M.J.: Distributed restoration strategies in telecommunications networks. In: IEEE ICC. Volume 1. (1994) 483–488