Fast Rerouting in ATM Networks: Pro-Active Search protocol

I. Lievens, T. Cattrysse, P. Demeester
Department of Information Technology, University of Gent
Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium
Tel: +32 9 264 33 16, Fax: +32 9 264 35 93
ilse.lievens@intec.rug.ac.be

Abstract

This paper introduces a rerouting algorithm for ATM networks - called the Pro-Active Search Protocol - which enables efficient rerouting in a simple and uncomplicated way. Therefor the protocol performs as many tasks as possible in advance to reduce the complexity. Basically this comes to working with shortest path algorithms based on a known network topology. In order to increase the rerouting performance this is combined with real-time actions after a rerouting trigger, by collecting accurate information about the network load. This is achieved by means of a simplified distributed approach.

Keywords
Network Availability/Reliability, Protocol Design

1 INTRODUCTION

In ATM Broadband Telecommunication Networks a wide range of services, each with particular demands and characteristics, will be offered to the user. An important issue in ATM Networks is the Quality of Service (QoS) guaranteed through negotiation contracts. The Quality of Service indicates the user's importance of the service, translated in terms of guaranties for delays, limits on cell losses, etc. A contribution to enabling the QoS guarantees is provided by mechanisms in the network which allow for protection against undesirable events,

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35388-3_42

H. R. van As (ed.), *High Performance Networking*

like network element failures or network area overloads, threatening the continuation and quality of the service provisioning, thus degrading contracts and decreasing customer satisfaction.

Intelligent routing aims at efficiently using the available network resources with respect to survivability, congestion control and possibly support of mobility. In all of these fields there is the requirement of finding alternative routes to replace interrupted connections, whether it is a network failure, a congested link or a user movement that caused the interruption. Considering the common point of finding alternative routes, it is more logic to indicate this with the term rerouting rather than restoration, since the latter limits the application area solely to dealing with network failures. Therefor rerouting will be used categorically in the rest of the paper to emphasise the key issue in this context: to find in an efficient way an alternative route to reroute an interrupted connection.

This paper introduces the Pro-active Search protocol, which tries to take advantage of static information present in the network and known in advance. At the same time the problem of acquiring accurate real-time information is tackled using some adjusted features of distributed restoration. The paper starts with some background on rerouting and some existing rerouting possibilities in Section 2. The new rerouting protocol is presented in Section 3, describing the general principles of the mechanism, followed by a description of the phases of the protocol. The protocol performance is discussed in Section 4 and Section 5 ends with a conclusion.

2 REROUTING TECHNIQUES

When looking somewhat more into detail in the survivability area, various mechanisms that deal with network failures have been proposed (see below). Their main characteristics, also applicable for the general rerouting problem, are important parameters which inherently influence the performance of the protocol: the moment when the alternative route is calculated or searched, and the control under which the restoration process takes place.

With pre-planned restoration, the routes are calculated in advance taking assumptions into account on network topology, network traffic, failure scenarios etc, and they are stored in databases. Upon the occurrence of a failure the routes only have to be looked up in the database, resulting in a fast restoration technique. However this only accounts for situations that have been anticipated in advance. Unexpected situations cannot be dealt with. Restoring at real-time on the other hand, implies that the route is only searched after the failure has occurred, resulting in a slower process, but with the advantage of a more accurate technique able to react at network changes even at failure time, making it a more robust technique. An efficient balance between the two extremes could result in a robust and time efficient technique.

Considering the control of the rerouting process, there can be an overall central control centre, coordinating every action of the process. The network nodes are unable to operate without its commands, making the overall mechanism highly vulnerable. With distributed control, every network node is provided with intelligence to carry out the restoration. Therefor they exchange information and take independent decisions to obtain an alternative route. There is no central overview of the network state, the nodes search a route by exchanging requests for spare resources. A problem with this distributed control is the increased complexity, mainly due to the high volume of messages flooding the network in parallel. Another drawback is the inability to influence the process in any way and its uncertain outcome. Again these are two extremes, an appropriate balance could result in a more efficient mechanism.

Fully distributed rerouting mechanisms have already been investigated (Han Yang, 1988) (Komine, 1990) (Struyve, 1996) (Lievens, 1996), with satisfying performance results. However the above mentioned drawbacks of complexity and unpredictability have led to the research for other techniques.

The Backup Virtual Path protocol, described in (Kawamura, 1992), is such a mechanism. Basically this technique provides for every working Virtual Path (VP) a backup Virtual Path, which takes over the working traffic if the working VP is interrupted for whatever reason. The route of the backup VP is calculated in advance and the VPI/VCI translation tables of the nodes on that route are already filled in. In contrast with protection, the backup VP does not take in any bandwidth in normal working conditions. When a failure occurs in the working VP, the backup VP is activated, bandwidth is captured and the backup VP takes over the traffic of the interrupted working VP. Variations are possible, but in its basic form the issue of unexpected events especially on the routes of the Backup VPs remains: if the backup VP is disturbed, the working VP cannot be restored.

Cooperations between backup VPs and a distributed rerouting protocol, which copes with these unexpected events, have been proposed as well (Chen, 1997), however by increasing the overall rerouting complexity.

Apart from these specific efforts in the restoration area, standardisation efforts on behalf of routing, signalling and rerouting in and between private ATM networks have been ongoing, being the Private Network-Network Interface (PNNI) (ATM Forum, 1996). In PNNI networks a hierarchical routing architecture is established by recurrently dividing the network nodes in Peer Groups with parent-child relationships. All the nodes are provided with databases, describing (different) parts of the network topology. The correct operation of PNNI is based on the information on topology, routes, links etc. stored in these databases, which have to be identical for nodes within one Peer Group. A lot of effort is put in ensuring that the information in these databases is accurate and up to date, using flooding and link state based protocols. Seen the efforts for ensuring that the databases are identical and up to date, the issue was raised to develop a protocol which could

take advantage of databases with pre-stored knowledge, while at the same time trying to include accurate information without lengthy updates.

3 PRO-ACTIVE SEARCH PROTOCOL

In developing the rerouting protocol the following goals have been taken into account throughout the design process. A very important issue has been to end up with a protocol that performs simple and efficient rerouting. Therefor it was important to consider what can be known and thus can be done in advance, however without degrading the performance because of inaccurate information. Therefor, some real-time actions with respect to required rerouting information have to be incorporated as well.

In the following sections, the general basic ideas are discussed first, followed by a more detailed formulation of the actual algorithm phases.

3.1 Background and basic principles

When considering rerouting in networks, many network aspects influence the process. Two of the most important and contributing issues are a network's topology on one hand and the network usage, i.e. the traffic or network load, on the other hand.

The topology of a network, which includes the nodes of the network and their interconnection by links, remains reasonably fixed and stable over a long period of time, certainly on the time scale important for rerouting of affected connections (i.e. seconds). The traffic in the network on the other hand, is flexible and subject to more frequent changes, as users will regularly start up new connections while other connections are torn down.

Since the topology is unlikely to change regularly on a short time basis (seconds to minutes), the topology can be regarded as a quasi-static given in the rerouting process. Therefor the rerouting protocol assumes that the network topology is known by the network nodes. This knowledge will be used to calculate candidate alternative routes in advance. Major changes in the topology, like the installation of new nodes or the upgrading of links, are distributed as updates to the nodes. This keeps them informed of the general 'static' network topology. Distributing topology information is a well-known issue and a common aspect in today's networks like for example the Internet; also in ATM PNNI routing the spreading of topology information is very important. It is emphasised that these updates only occur in an orchestrated way at regular intervals, after a major change in topology. More detailed knowledge on how these updates are performed is irrelevant with respect to the further flow of the protocol. It is noted here that an unexpected event like a network element failure is not considered as a major topological change. From this point on the overall working network topology is assumed to be known.

The traffic in the network on the contrary, is more likely to change at much shorter time intervals than the topology. This makes the network load a real-time and more uncertain aspect and it is thus considered as a dynamic issue. Therefore it is assumed that the network nodes are not aware of the current overall network traffic.

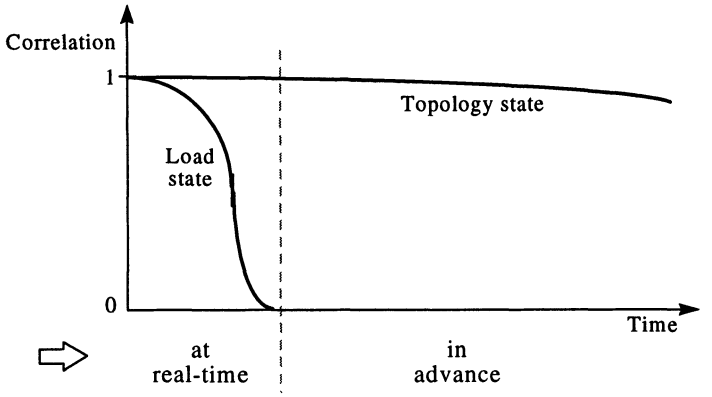

Figure 1 Network state correlation in time.

A short intuitive note on the accuracy of network information, seen from the viewpoint of the elapsed time between getting the information and actually using it, can be found in Figure 1, which shows the qualitative correlation between knowledge of the current network state and knowledge of the network state at some point is the future. The network state for the rerouting purposes described here, is divided into topology state and load state information. Given the highly static character of the topology, the topology state correlation degrades only slowly: knowledge of the current topology will most likely imply knowledge of the future topology. The network traffic is dynamic, resulting in a rapidly decreasing correlation. Knowing the network traffic now, does not imply that the traffic is known at a moment in the future, when this information might be needed for rerouting. This implies that calculations on topology can be carried out in advance, while accurate information on traffic must be collected in real-time.

When the network is confronted with unexpected changes in topology, caused for example by failures of network elements, this can interrupt working connections and it is important to restore these connections as fast as possible, in this case by finding alternative routes on which the connections can be provided again. That is dealt with by the real-time aspect of the rerouting protocol. It takes advantage of the knowledge of the network topology by using pre-calculated routes. Network traffic information is acquired at real-time using these routes. This deals with obtaining information on the bandwidth still available on links.

3.2 Phases in the algorithm

Pre-rerouting phase: calculation of the routes

The purpose of this pre-rerouting phase is to provide each network node with a number of paths between him and every other network node, thus taking advantage of the knowledge of the network's topology. These paths are to be used in the next phase, where the actual rerouting will take place upon the occurrence of a rerouting event.

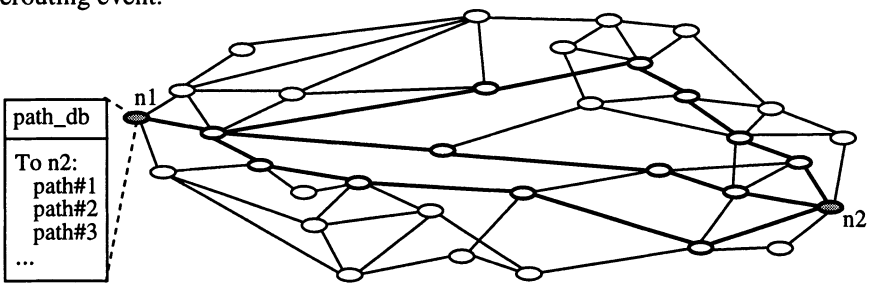

Figure 2 Pre-calculated paths.

A number of paths between every pair of nodes has to be calculated, which are stored in a database in the node (Figure 2). The paths can be determined because the topology of the network is assumed to be known to every network node. For the computation of the paths, an algorithm is required which is able to determine a number of paths between a pair of nodes, based on the topology of the network in terms of nodes and links. This algorithm can be a static and central one.

It is preferable that the routes are efficient in terms of length, cost, etc. Therefor the Pro-Active Search Protocol uses an algorithm that calculates the k shortest paths between every node pair in a network. The qualification shortest is important, since alternative routes should be efficient in terms of spare resources and since these paths will eventually be used in a time sensitive environment: the sooner an alternative route is found, the less information is lost.

Algorithms for finding k shortest paths between two nodes in a network can be found in literature. The algorithm used here is the algorithm of Yen, which finds k shortest loop-less paths in a network. The full description can be found in (Yen, 1971). It uses an arbitrary iteratively applied shortest path algorithm, in this case Dijkstra's. Furthermore Yen's algorithm avoids loops in the paths, basically avoiding that nodes are present in a path more than once. This feature is important for the efficiency of the rerouting. As far as computation time is considered, Yen's algorithm has an upper bound which changes only linearly with the number k.

It is however important to note that basically any algorithm can be used here, which is able to calculate k shortest paths from a network topology in which the nodes, the links and their costs are given. A short overview is given in (Shier, 1997).

In moderate sized networks it is feasible to calculate k paths between every pair of two nodes in the network, in other words to have every node store k paths to every other network node. When networks grow larger in size, this might become unpractical. An option is then to limit the pairs in geographical distance and only calculate paths between any pair of nodes not too far away from each other. Another solution can be found in hierarchical networks, like for example in PNNI networks [7]. There the paths can be confined within a peer group, limited in size anyway, and calculated between any pair of peer nodes.

It is not considered efficient to store all the paths of the entire network or network group in some central location, which is consulted after a failure and which downloads the required paths. This is considered to be too vulnerable and time consuming. In stead every node has its own database, in which paths from this node to all or some other nodes are stored. When required, the paths are immediately available.

The paths have to be computed prior to a rerouting event, explaining the term pro-active. Whenever a major topology update is issued, the routes must be recalculated, to ensure the effectiveness of the paths. This recalculation - using the same k shortest path algorithm as before, but now on the updated topology - is a background process, running in parallel or more accurately in between rerouting processes. Possible inconsistencies of databases between nodes are here not as important or threatening as for example in PNNI. Because a node knows the entire path, it must not rely on a next intermediate node to determine the appropriate next hop in the path. The network nodes don't need identical databases to ensure a correct result. Moreover, since a number of paths are pre-calculated, the impact of one invalid route is low, at most only slightly decreasing the performance.

Rerouting phase

This is the part of the rerouting protocol that executes real-time actions in order to collect real-time link information. This will result in obtaining valid alternative routes, which will actually replace the failed connections. The required real-time information on the load of the network is collected by sending out search messages - so-called Courier messages - in a controlled way.

This real-time rerouting part is triggered by a so-called rerouting trigger event. This is an unexpected network event, like the failure of a network element, interrupting working connections. Also a network link becoming overloaded with risk for congestion, can act as a trigger to search alternative routes avoiding this crowded link.

As far as the rerouting is concerned, there is the choice to reroute on link or on path basis (Figure 3). With link rerouting, only the unavailable link will be replaced by an alternative route, the undamaged parts of the connections using the failed link are kept. In case of path rerouting, the entire connection(s) using the failed link will be rerouted. The latter needs some extra complexity because the connection endnodes must be notified of the rerouting and the capacity along that

connection on working links must be released. This requires some extra delay as compared with link rerouting, which is a local process and generally faster. However the total alternative route in link rerouting might be less efficient.

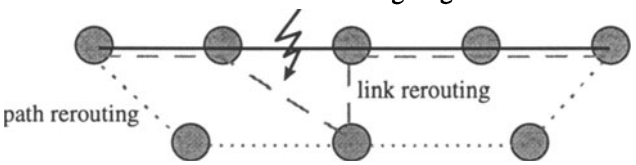

Figure 3 Link and Path rerouting.

The nodes between which the alternative route is searched - whether it are the link adjacent nodes or the connection end nodes - are called the restoration node pair. They will start up and eventually terminate the real-time process. In this protocol, the Sender-Chooser technique (Grover, 1987) is used, in which one of the restoration pair nodes starts sending out Rerouting Messages, while the other one will eventually choose the alternative routes found during the rerouting process. They are called Sender and Chooser node (Figure 4). The arbitration of Sender and Chooser is mostly based on node IDs. It should be noted that in case of path rerouting different Sender-Chooser pairs will be active in the network, one for each failed connection. This can complicate the process.

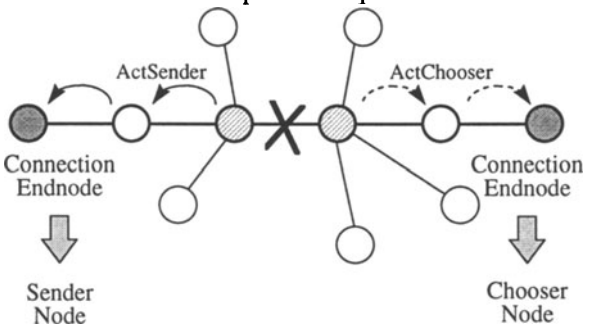

Figure 4 Sender-Chooser arbitration.

Actions of Sender and Chooser nodes

The Sender node consults its database for the k pre-calculated shortest paths to the Chooser node. On each of these k paths, the Sender node sends one Courier Message to the first node on these routes, which represent the possible alternative routes (Figure 5). Every Courier Message collects and stores real-time information about that route on its way to the Chooser node. Basically this deals with the available bandwidth of the links on the route.

The Chooser node will initially just wait, until Courier Messages arrive from any of the routes.

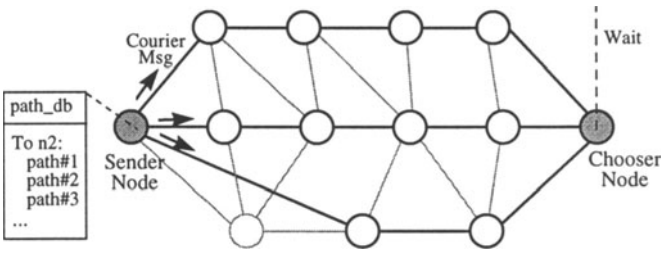

Figure 5 Actions of Sender and Chooser.

Controlled flooding: forwarding of Courier Messages

The Courier Messages are forwarded from node to node along their specific route, meanwhile collecting appropriate real-time link information (Figure 6).

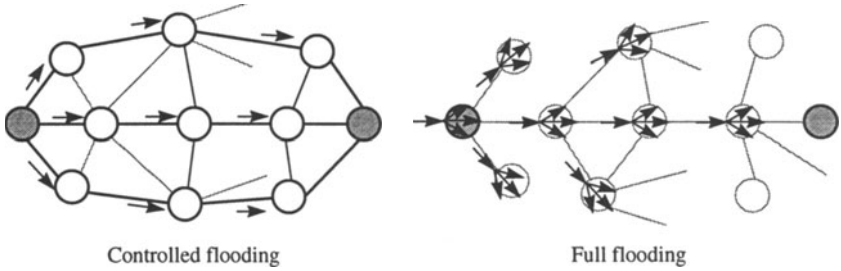

Figure 6 Controlled vs. full flooding.

This phase has characteristics of a distributed restoration algorithm: the network nodes perform the required rerouting actions without commands from some central control unit. Some of the major drawbacks of fully distributed algorithms however - being the uncontrolled flooding of restoration messages, the resulting complexity and the unpredictable outcome of the restoration process - is avoided by the controlled forwarding of Courier Messages. The messages are not flooded to all neighbours, which could create an avalanche of parallel messages roaming in the network, but they are only sent on specific and fully specified paths. Also the network area extent which is visited by the messages is controlled: only the links of the pre-calculated routes will carry rerouting messages. This implies that the network operator has more control on the alternative routes to be found, while still having the advantage of self-healing facilities in the nodes.

The rerouting messages do not reserve bandwidth along the route. They only collect the real-time load information and bring this information to the Chooser node. This strategy avoids complex release protocols at the end of the process, required to release bandwidth previously reserved for a particular failure but which is no longer needed because, for example, other routes have been found.

A Courier Message will only be forwarded on the next link of the route if there is still some available bandwidth on that link. This implies that a Courier Message encountering a link without spare capacity is not forwarded (Figure 7). As far as

that particular route is concerned, the rerouting process stops: this is not a viable alternative route. No cancel or release messages are required.

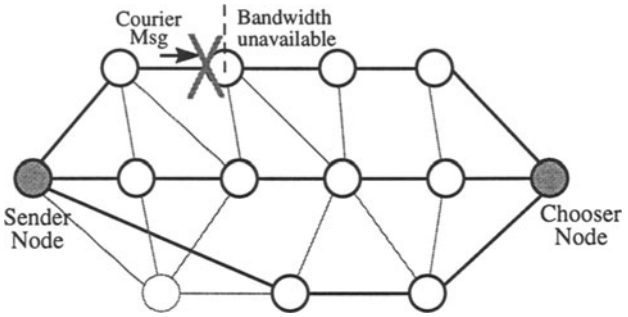

Figure 7 Pre-calculated path unavailable.

Actions of the Chooser node

Assume that a Courier Message arrives in the Chooser node. With the collected information about the links' load in the message, the Chooser node can build up an overview of used and available capacity on the routes between Sender and Chooser. On that basis the Chooser can observe whether a route has available capacity and decide to take a route as actual alternative route. The Chooser consequently adjusts its capacity allocation overview to indicate that some of the available capacity is taken by an alternative path. This is mainly important when the k routes have some links in common and conflicting situations in the allocation of the available capacity can occur. The choice of an actual path by the Chooser is done on first-come first-take basis. It is also possible to wait for a certain time period and then to choose between the arrived candidates the route with the largest sufficient bandwidth.

Route confirmation

When a route is chosen, the Chooser node sends back a Confirmation Message on that route towards the Sender node. This message will ensure that the nodes on the route will adjust their routing tables and actually reserve the capacity for that route. In that way, the message is rippled back to the Sender node, which then knows that a valid alternative route is found.

It is possible that on the way from Chooser to Sender, a link is encountered where the capacity assumed to be free is no longer available because for example another rerouting process has taken in that bandwidth. In that case a Cancel message is sent back to the Chooser node and, if available, another route is taken in stead.

The rerouting process is terminated when all failed capacity can be rerouted along alternative route(s) or when there is no more available bandwidth on the k routes. In the latter case some of the failed capacity cannot be rerouted.

Remark on the k shortest paths

As can be derived from the previous sections, the nature of the pre-calculated paths highly contributes to the success chances of the protocol. At the end of the rerouting phase, some of these paths will be used as the actual alternative route(s) for the affected connection. It is obvious that these paths will seriously affect the final rerouting performance.

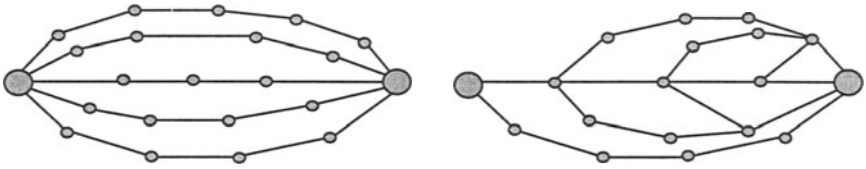

Figure 8 Nature of the pre-calculated paths.

The number of pre-calculated paths is an important parameter in the rerouting process, influencing the performance and the rerouting capabilities of the protocol. The number k must be large enough to have reasonable chances that a feasible route can be found eventually.

The paths must also be efficient in terms of the used links. This is concerned with how the k calculated paths between a node pair are related to each other, with respect to using the same links and nodes (Figure 8). With Yen's algorithm it is highly probable that the k shortest paths between two nodes share a number of links and nodes. If such a shared link has little available capacity, this means that immediately a number of the k paths become unavailable as alternative routes, degrading the rerouting chances. The number k does not directly refer to the number of disjoint paths between a node pair. However it is important to realise that fully disjoint paths are not required for this protocol, simply a number of pre-calculated routes covering a certain area which can be searched in a controlled way at real-time. Besides, the connectivity degree of the network limits the maximum number of disjoint paths due to the limited number of outgoing links out of a node.

An simple extension has been added to the Yen algorithm to avoid that in an extreme case a link is used by all of the k pre-calculated paths. Basically a threshold is used to set the maximum number of paths that can use a same link. This increases the rate of success, since this one link is a possible bottleneck to the rerouting process. The protocol leaves ample space for other algorithms that can calculate k paths between two nodes.

The use of pre-calculated paths also allows for a more particular choice of the routes. In stead of just calculating the k shortest paths, one could provide the databases with paths that cover only a certain network area, recommended for rerouting. This would allow network operators to enforce a certain rerouting strategy.

4 PROTOCOL PERFORMANCE

In order to verify the behaviour of the protocol as well as to assess its performance in terms of rerouting capacity, the protocol was described in the standard language for protocol description, being the Specification and Description Language SDL (Z.100, 1988), by using the SDL Design Tool SDT, which provides the means to describe and simulate communication protocols.

Each node is assigned a model of a FIFO queue and one processor. It takes 10 ms to read in and process an incoming message and 10 ms to generate an outgoing message. The simulations were performed on a realistic 32 node network, provided with enough spare capacity to enable the rerouting of all single link failures.

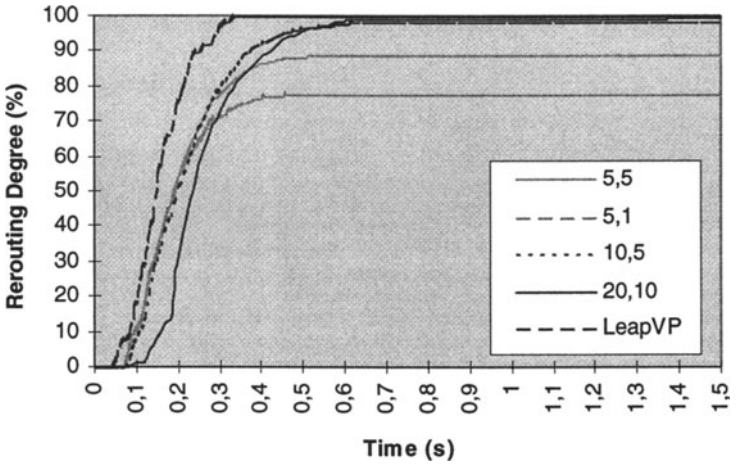

Figure 9 Single link failures (k, threshold), link rerouting.

The average rerouting degree for link rerouting of all single link failures in the sample network as a function of the elapsed rerouting time in Figure 9. When k is small, there is usually not enough total spare capacity to reroute all failed connections. The combination (20,10) showed the best degree. When comparing with a distributed technique simulated with the same parameters (Lievens,1996), it shows that indeed the in advance actions of this protocol do give good results.

The same network was simulated for single link failures applying path rerouting. The obtained results are significantly less than with link rerouting. This is due to the fact that with path rerouting, different simultaneous search processes are ongoing in the network, one for each connection that used the failed link. These simultaneous processes compete with each other for spare capacity, thus explaining the performance degradation.

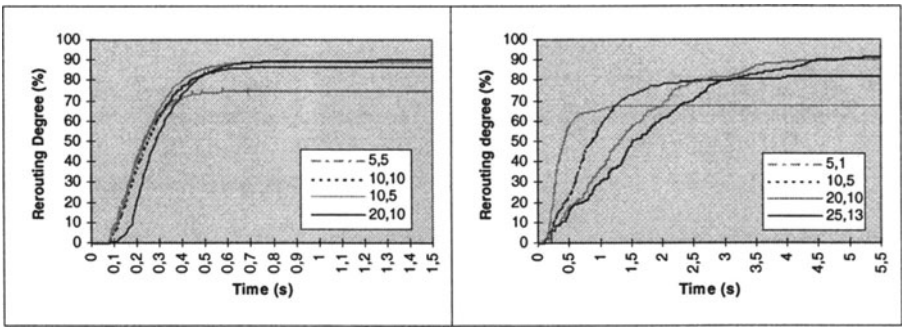

Figure 10 Double link failures; node failures.

Simulations were performed for node failures and double link failures (Figure 10), on the same network with fixed spare capacity, resulting in satisfying performance. For the node failures, path rerouting was used, explaining the longer time needed to achieve rerouting.

5 CONCLUSION

In this paper a new rerouting protocol is introduced for ATM networks. The Pro-Active Search protocol combines real-time action with path calculations carried out in advance. This is based on the assumption that a network's topology is quasi-static. Therefore this topology is assumed to be known by the nodes, and they take advantage of this by calculating shortest paths between them and every other network node. The result is that each node knows k paths to another node, as far as the general topology is concerned. The network's traffic is assumed to be dynamic and flexible; this is considered as unknown by the nodes. When rerouting is required, the nodes will gather accurate, real-time information of the load on the pre-calculated paths by sending a limited number of Courier messages on each of the k paths. In this way speed and accuracy are combined, resulting in a fairly simple protocol with the advantages of pre-planned restoration together with those of real-time distributed restoration.

Acknowledgments

This work has been supported by the Flemish Government through the IWT project ITA/950214/INTEC.

6 REFERENCES

Han Yang, C. and Hasegawa, S. (1988) FITNESS: Failure Immunization Technology for Network Survivability. *Globecom '88*, 1549-54.

Komine, H., Chujo, T., Ogura, T., Miyazaki, K. and Soejima, T. (1990) A Distributed Restoration Algorithm for Multiple-link and Node Failures of Transport networks. *IEEE 1990*, 459-63.

Struyve, K., Demeester, P., Nederlof, L. and Van Hauwermeiren, L. (1996) Design and Evaluation of distributed link and path restoration algorithms for ATM meshed networks. *Proceedings International Zurich Seminar on Digital Communications, ETH Zurich, Switzerland*, Vol. 1044.

Lievens, I. and Demeester, P. (1996) The use of distributed restoration in intelligent routing. *Proceedings of IEEE Fourth Symposium on Communications and Vehicular Technology in the Benelux.*

Kawamura, R., Sato, K. and Tokizawa, I. (1992) Self-healing ATM Networks Based on Virtual Path Concept. *Networks*, 129-34.

Chen, S. and al. (1997) An Integrated Restoration Approach (IRA) in the ATM Network. *Globecom 1997*, 1388-92.

The ATM Forum Technical Committee. (1996) Private Network-Network Interface Specification Version 1.0 (PNNI 1.0). *af-pnni-0055.000*, March 1996.

Yen, J.Y. (1971) Finding the k shortest loopless paths in a network. *Management Science*, Vol. 17, 712-16.

Shier, D.R. (1979) On Algorithms for Finding the k Shortest Paths in a Network. *Networks*, Vol. 9, 195-214.

Grover, W.D. (1987) The Self-Healing Network: a fast distributed restoration technique for networks using digital cross-connect machines. *Globecom 1987*, 1090-5.

ITU Recommendation Z.100 SDL-88. (1989) Functional Specification and Description Language.

7 BIOGRAPHY

Ilse Lievens graduated in Electrical engineering at the University of Gent in 1994. Her graduation thesis ("The influence of restoration strategies on the performance of wavelength multiplexed networks") dealt with static and centralised restoration in WDM networks. Since September 1994 she is working in the Broadband Communications Networks Group as a research associate and is preparing a Ph.D. Her research interests include survivability issues in ATM networks. In this context she was involved in the RACE II project IMMUNE, and the set-up of the PANEL project. Her research is extended towards intelligent routing, focusing at a common rerouting strategy for survivability, congestion and mobility.