A Fast and Efficient Backup Routing Scheme with Bounded Delay Guarantees

Enrique Hernández-Orallo
Dto. Informática de Sistemas y Computadores (DISCA)
Universidad Politécnica de Valencia. Camino de Vera s/n
Valencia, SPAIN
ehernandez@disca.upv.es

Joan Vila-Carbó
Dto. Informática de Sistemas y Computadores (DISCA)
Universidad Politécnica de Valencia. Camino de Vera s/n
Valencia, SPAIN
jvila@disca.upv.es

Abstract — Reliable transmission is essential for several real-time applications. Backup Channels introduce the notion of availability at the cost of increasing the use of network resources. However, this over provisioning of resources is potentially wasted, since packet delays are usually lower than the required end-to-end channel delay. The goal of this paper is to present a new scheme for obtaining the primary and backup paths maximising the admission of channel in a network

This paper uses a new failure detection scheme for selecting the primary and backup paths denominated Proactive Backup Channel. This scheme is based on activating the backup channel before a fail is produced. The experiments show that using this new scheme the admission rate (the number of channels a network can accept) is improved considerably.

The paper also presents the application of this scheme to IP networks using TCP trunking.

Quality of Service; Dependable Networks; Routing; TCP trunking.

I. INTRODUCTION

Several distributed real-time applications need a reliable transmission: telesurgery operations, real-time guidance and control, critical experiments in dangerous environments (nuclear and biological), tele-conference, etc. Service disruption for even a short duration could be catastrophic for this type of applications. The real-time requirements and dependability issues often conflict, resulting in a trade-off situation [1]. For example, increasing the reliability of a real-time transmission using packet retransmissions may increase the end-to-end delay so a packet misses its deadline. In many ways, the concept “Quality of Service” (QoS) already represents the concurrence of many fault-tolerant and real-time services [2]. That is, real-time distributed applications require Quality of Service guarantees on timeliness of message delivery and failure-recovery delay.

However, the problem of dynamically routing QoS paths with restoration and bounded delay has not been studied extensively. In this paper we present a new scheme for the problem of setting up primary and backup paths with a guaranteed bandwidth and bounded end-to-end delay. Previous works [1,2,3] only focus on bandwidth, but for distributed real-time application is necessary to guarantee a maximal network delay.

This leads to a new QoS routing problem: it is necessary to dynamically obtain the route of the primary path and the backup path in order to setup a new dependable channel. Simultaneous routing of both paths guarantees that if the primary path fails then the backup path will have sufficient resources and the end-to-end delay will be guaranteed. This routing problem appears in many networks contexts, for example in MPLS networks or wavelength (lambda) switched paths in optical networks.

The goal of QoS-based routing is to find a path that meets the requested QoS requirements optimising the network utilisation [4]. Usually, standard routing algorithms are single objective optimizations, that is, they try to minimize one cost function [4,5,6]. Multiple objective path optimization is a more complex task [4,7], and is known to be a NP-complete problem so several heuristics or polynomial approximations has been proposed in order to obtain a path in a practical bounded time [8,9,10,11,12,13,14]. Since these algorithms are computational complex and do not scale well, precomputation-based schemes have been recently introduced [5,15].

The contributions of this paper are twofold: first we introduce a new failure detection scheme and second a fast algorithm for obtaining the primary and backup paths. The goal is to obtain a fast algorithm (necessary in real-time environments) that maximises the number of channels that the network can admit.

The mechanism for failure detection is based in timeouts. The latency of failure detection is a key aspect in providing an uninterrupted service (availability). Two ideas are the basis of this scheme: first, the end-to-end probability delay function (PDF) has a long tail (a great percentage of the packet delay is very far from its deadline) [16]. Second, the longer the failure latency time is, the lower the primary path delay requirements are (and this implies that the network can admit more channels).

This scheme is a slightly variation of the scheme for fault detection introduced by the authors in [17,18] called Proactive Backup Channel (PBC). Proactive Backup Channel reduces failure latency by “suspecting” a failure before it occurs (it is proactive versus the classical reactive approach).

1 This work has been supported by the Spanish Government Research Office under grant TIN2005-08665-C03-03
The goal of the routing algorithm is to find primary and backup paths with a maximum delay constraint and enough resources to meet the channel requirements. This problem is easily transformed in finding first, a primary path, and then a secondary path with no common links with the primary path. The problem of finding a route, in its general form is a NP-hard problem. We present a different approach to resolve this problem with low polynomial cost.

In section IV we evaluate the performance of this new approach and the result shows that this new scheme admits more dependable channels than the classical approach. Finally, we detail the implementation of this scheme using TCP trunking [27]. TCP trunking is an easy way to implement this scheme over existing IP networks.

II. PROACTIVE BACKUP CHANNELS

There are several approaches for network rerouting. These can be classified into link-based schemes and path-based routing schemes [3]. The scheme presented in this paper is a path-based routing scheme. There are two main approaches for path-based dependable channels:

- **Multiple copy** (MC) [19,20]: several redundant real-time channels are set-up through disjoint routes and packets are sent simultaneously through all routes. There is no need for a failure detection mechanism: the scheme works as long as the receiver gets at least one of the replicated packets. This is a costly technique (all the channels are permanently used) but it provides uninterrupted service in spite of failed routers.

- **Reactive Backup channels** (RBC) [21]: a primary channel and a backup path are required, but the backup path is only set-up upon a failure of the primary path. That requires a failure detection mechanism. Since the backup path is not used until a failure occurs, this technique saves unnecessary bandwidth utilisation, but an additional delay is introduced for setting up the backup path. Besides, it may occur that there exist no enough resources when trying to set-up the backup path.

Reserving resources in advance for the backup path can solve some drawbacks of the second approach: this would improve the set-up time and avoid the lack of enough resources. Resources for the backup path are reserved but not used in absence of failures, so they can be used to transmit non-critical traffic.

Some variations of the Reactive Backup Channels have been introduced. In [22] the authors introduce a scheme to use the unused spare resources (in some aspect it is like overbooking). A similar approach is used in [23], using a probability model of the activation of the backup path in a node. A variation of the backup scheme is the segmented Backup Scheme [24], in which backup paths are provided for partial segments of the primary path rather than for its entire length.

A crucial aspect of the backup channel scheme is the failure detection mechanism. The most common proposals for failure detection are based on heartbeats ("I am alive" or "ping" messages). Heartbeats can be used to detect failures of neighbour nodes, or in an end-to-end fashion. An important problem is failure detection latency, especially in the end-to-end detection technique. This latency can be a serious obstacle for applying these techniques, since it makes the delay requirements of the backup path very demanding in order to keep the service uninterrupted, and that causes the corresponding resource reservations to be sometimes unacceptable.

In this paper we introduce a variation of the Proactive Backup Channels (PBC) introduced in [16]. The Proactive Backup Channel scheme is a new failure detection aimed to achieve an efficient channel resource reservation by reducing the failure detection latency. It is based on “suspecting” a failure when the packet delay through the primary path is close to the maximum guaranteed delay. This delay can be mainly due by two reasons: the first reason is network congestion and the second is a packet loss. Therefore, in the PBC scheme whenever a failure is suspected, the backup path is activated although the primary path would not be discarded. This could yield unnecessary backup paths activations due to inaccurate failure detection (false failures due to network congestion), but it has the advantage that the delay requirements of the primary path are not so restrictive. That allows reducing the required resource for the primary and backup paths.

As detailed in [18] the efficiency of the Proactive Backup Channel is based on the following two typical network behaviours:

- **End-to-end delay distribution**: Packet end-to-end delays follow the typical distribution of Figure 1. More important is that real delays are very far from the maximal delay. Therefore, a given activation delay that corresponds to 99% of the packets can be easily derived from this curve. If a packet time is higher than the activation delay the client starts to set-up the backup path.

- **Network resource utilisation increases when the required end-to-end delay is reduced.**

![Figure 1. End-to-end delay distribution](image)

The second condition can be used only with individual or aggregate flows. In order to work with network traffic the condition can be easily reformulated as:

- The lesser the required delay the lesser the number of channels accepted. This is an obvious fact in network
reservation. If we have several flows with very low end-to-end delays it will be hard to find a path that fulfils the delay.

The objective is to find a dependable channel, that is, a primary and backup path, guaranteeing a maximal end-to-end delay. If the primary path fails then the backup path is used but always guaranteeing that transmission delay is bellow the required deadline.

Let $d_f$ be the failure detection time and $d_A$, $d_B$ the end-to-end delay assigned to the primary and backup paths (see figure 2). In the RBC approach $d_A=d_f$ so the maximum delay experienced by the first packet retransmitted through the backup path upon a failure can be expressed as $d_{\text{total}}=d_A + d_f + d_B$ where $d_f$ is the setup time for the backup path. Consequently, $d_B=d_{\text{total}}-d_A-d_f$. In the PBC approach $d_f < d_A$ so the end-to-delay can be now expressed as $d_{\text{total}}=\max(d_A+d_f+d_B)$. This way, the upper bound for $d_A$ could be $d_{\text{total}}$ while $d_B$ could be chosen as $d_{\text{total}}-d_f-d_f$. The fact that $d_f$ is $d_{\text{total}}$ implies an important reduction in delay requirements for the primary path. Then, the path selection will depend on the selection of $d_f$. This value must be carefully selected depending on the end-to-end delay distribution (this is an engineering decision). A lower value of $d_f$ implies a higher $d_B$ value; that would lead to a reduction in the delay requirements for the backup path, but would produce false failures. For example, for the distribution of Figure 1, with a maximal end-to-end of 400 ms and setting $d_f = d_{\text{total}}/2 = 200$ms this would imply that 0.1% of the packets would produce a false failure activation. If we use $d_f$ as $d_{\text{total}}/3 = 133$ms the false failure ration will be 5%. As will be shown in section 5, the set-up time for the backup path ($d_f$) is insignificant when it is compared with the total delay ($d_{\text{total}}$). Therefore, to simplify, $d_f$ is assumed to be 0 for the rest of the paper.

The first channel with $d=8$ is accepted in both schemes. But the second channel is accepted in the PBC scheme but not in the RBC scheme. The reason is that the required delay in the primary and backup path is 2s and only one path (the primary or the backup) can be admitted. As a result the channel is rejected. Instead, in the PBC scheme the primary path has a delay of 4s and can be routed using $P2$; the backup path is $P1$.

<table>
<thead>
<tr>
<th>Channel</th>
<th>PBS</th>
<th>RBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>N</td>
<td></td>
</tr>
</tbody>
</table>

Although this is a very simple case, the simulations of section 4 will show that the PBC scheme accepts more flows than the PBC scheme and makes network utilization higher.

In summary, the proposed scheme is a trade-off to allow inaccurate failure detection in order to minimise latency, increasing, thus, the network utilisation.

III. A FAST PATH ROUTING ALGORITHM

This section describes a two path selection algorithm, which for a given network topology, link metrics and flow QoS requirements, computes a primary and a backup path with a bounded end-to-end delay minimising the network utilization.

A. Metrics and network model

Selecting a path that can satisfy the QoS requirements of a new flow is based on the knowledge of the flow's requirements and information about the availability of resources in the network [5]. This information is usually defined as metrics on which the routing algorithm works. The network metrics must be periodically updated and distributed to the ingress routers in order to make an accurate path selection.

The network model is the following: $G(V,E)$ represents a network, where $V$ is the set of nodes (vertices), $E$ is the set of
links (edges), \( N \) is the number of nodes and \( M \) the number of links.

Each link \( E_{ij} \) represents a connection from node \( i \) (\( V_i \)) to \( j \) (\( V_j \)) and is characterized by the following values (the link metrics):

- Bandwidth (\( R_{ij} \))
- Available bandwidth (\( A_{ij} \)). These values are updated when a new flow enters (leaves) the link.
- Maximal delay \( D_{ij} \). The maximal delay of the link (includes queue delay and link propagation delay).
- The flow to be routed is characterized by the following values:
  - A source node \( s \) and terminal node \( t \).
  - Required Bandwidth reservation \( r \).
  - A maximal end to end delay \( d \).

Furthermore, a path \( P_{st} \) is defined as \( P = \{s = V_{i_0}, V_{i_1}, ..., t = V_{i_l}\} \) such that \( \forall 0 \leq i < l-1, E_{i,i+1} \in E \) and \( H \) is defined as the number of hops of the path.

As one objective is to reduce the utilization of the link bandwidth in all the nodes the cost function can be expressed as an exponential function of the bandwidth utilization in the links along the path:

\[
C(P) = \sum_{E_{ij} \in P} (U_{ij})^{e} U_{ij} = \frac{R_{ij} - A_{ij} + r}{R_{ij}} \quad (1)
\]

The greater the coefficient \( e \) the greater the cost when utilization is near to 1, avoiding the selection of paths that has links with high loads.

**B. Path computation algorithms**

Finding a primary and backup path with disjoint channels can be easily reduced to finding a single path [25].

1. Obtain the primary path with a delay \( d_A \).
2. Delete the links used in the primary path
3. Find a backup path with delay \( d_B \)

Thus, the problem is reduced to finding a path with a maximum deadline of \( d \). Following the notation in [4], the algorithm has a link-constrain, a path-constraint (maximum delay in the path) and an optimization function. This problem is known to be NP-complete. Formally, the problem can be formulated as follows:

Given a network \( G(V,E) \) with link parameters \( R_{ij} \) and \( D_{ij} \), and a new flow connection \( i \) with source \( s \) and destination \( t \) and \( r \) and \( d \) parameters, find a feasible path \( P^* \), with the following constraints:

1. \( r \leq A_{ij} \quad \forall E_{ij} \in P^* \)
2. \( \sum_{E_{ij} \in P} D_{ij} \leq d \)

and minimizes the cost function \( C(P^*) \).

If all the links \( E_{ij} \) for which \( r > A_{ij} \) are eliminated from the graph, then this problem has been reduced to the path-constrained path-optimization routing that is also known as the RSP(Restricted Short-Path) problem. This is a NP-complete problem and several approaches have been devised in order to resolve the problem in a bounded time. A more extensive survey of solutions can be found in [4, 12, 26].

In this paper we use the RICP (Restricted Iterative Constrained Path) algorithm presented in [26]. This algorithm obtains a feasible path with low computational cost. The idea is to convert the path-optimization condition (the cost optimization) in a link-constraint condition so it can be resolved in polynomial time. Therefore, before selecting a path, we can check if the utilization of all the links in the network is less than a given limit value \( L_u \), and if not, delete them. With this new network, we find the path with the needed bounded delay (if it exists). This step can be repeated for several limit values from a low utilization limit to a high utilization limit, until a path is found. The computational cost of this algorithm is: \( O(N \times M \times S) \) where \( S \) is the size of the set of limits \( L_u \). As proven in [26] the value of \( S \) does not depend on the network size and is very low (10 or less), so this algorithm is very fast.

The algorithm for finding the primary and backup channel is presented in figure 4, using the RICP algorithm.

```
Algorithm RICP(G, F, SL)
Input:
G, F, SL

Output:
P \quad 'Path found or void

begin
  G' = G
  for all \( E_{ij} \) in \( G' \)
  if \( r > A_{ij} \) then
    remove \( E_{ij} \) from \( G' \) 'First Condition
    \( U' = (A_{ij} + r) / R_{ij} \);
  if \( U > L \) then
    remove \( E_{ij} \) from \( G' \)
  end for
  P = find_shortest_path(G')
  if delay(P) < d then
    return P 'Path found!!!
  end for
end;
```

```
Algorithm TwoPaths(G, F, SL)
Input:
G(V,E,R,D,A)

Output:
P, P_b 'Primary and backup paths

begin
  \( P = \text{RICP}(G,F, SL) \);
  G' = G
  for all \( E_{ij} \) in \( P \)
    'Remove all links primary path
    remove \( E_{ij} \) from \( G' \)
  end for
  P_b = \text{RICP}(G',F, SL) ;
end;
```

Figure 4. Algorithm for finding the primary and backup paths
A. Admission and utilization experiments

The goal is to evaluate the number of dependable channels that can be established using the PBC schemes versus the RBC scheme. We selected two metrics: the percentage of rejected flows from the total of flows generated (the blocking ratio) and the link utilisation.

We assume online routing. Flows arrive to the network. Each request has an ingress node, an egress node, a required bandwidth and a maximal delay. The goal of the online routing algorithm is to obtain a primary path and a backup path for every flow.

The traffic is generated based on an offered bandwidth parameter $O_o$ that is defined as the mean bandwidth of the workload. Then, flow arrivals are generated according to the Poisson process with a parameter $\alpha$ and their durations are exponentially distributed with mean $1/\beta$. The average number of flows (the ratio $\alpha/\beta$) is obtained as the quotient of the offered bandwidth ($O_o$) and the mean bandwidth reservation of the loads. The flow parameters are generated in this form: $r$ is uniformly distributed in the range $[r_{lb}, r_{l}]$ and the delay $d$ is uniformly distributed in the range $[d_{lb}, d_{l}]$. Additionally, configurable load traffic is introduced in the links in order to load the network. This load is introduced as a portion of the link capacity.

The simulator tries to establish a dependable channel for the flow finding a path for the primary and backup channel. The delay allocation follows the equations of Figure 2. The failure detection time is set as $d/2$ so the delay of the backup path is $d/2$ for both schemes.

The first experiment used a very simple network. It is a grid of 4x4 nodes and is detailed in figure 5. All the links have a nominal bandwidth of 100Mbps and link delay of 0.01s. The traffic load parameters were the following: $r_{lb}=0.01$Mb/s, $r_{l}=1$Mb/s, $d_{lb}=0.06$s, $d_{l}=0.2$s, flow duration $1/\beta=100$s and total simulation time=86400s (24 hours). The source and destination nodes were randomly selected from the edge nodes. The minimum delay (0.06s) is selected as twice the minimum path delay (3 nodes=0.03s) in order to accept the backup channel ($d_{lb}=d_{l}$).

Figure 6 shows the blocking ratio between 50Mb/s and 500Mb/s using three network loads (0,25%,50%). The results are very clear: the PBC scheme admits more flows than the RBC scheme. This is more evident when network load is high. The difference between the two schemes is reduced when the offered load is very high. The reason is very simple: the blocking ratio is very high and practically all flows are rejected. But this situation reflects network congestion or insufficient network provisioning. Therefore, for operational blocking ratios (less than 0.2) the results show that the PBC scheme can admit about 10% more channels than the RBC scheme with no network load, about 40% more channels with 25% of network load and about 30% with 50% of network load.

![Network 1: Grid network](image)

**Figure 5.** Network 1: Grid network

![Blocking ratio in Network 1](image)

**Figure 6.** Blocking ratio in Network 1

Figure 7 shows the global network utilization (that is, the mean of the utilization of the network links). The results show that the utilization is greater for the PBC scheme.

We repeated the experiments with different load parameters and a variety of synthetic topologies generated with the Brite network generator following the Waxman model [20]. The results are very similar than the previous ones; the PBC scheme admits more channels than the RBC scheme. The difference between the admission rates of PBC and RBC depends mainly in two factors: the network characteristics and the required delay. The rest of the load parameters have low influence on the simulations. These dependences are logical, because the network topology is essential in determining if a dependable channel is admitted.
Therefore, the second experiment used a network based on the topology of the GÉANT network (as known in April 2004). GÉANT is a pan-European multi-gigabit data communications network, reserved specifically for research and education use. The network is detailed in Figure 8 and has 33 nodes and 47 links. The link delays values are approximated based on the propagation delay: ~5ms/1000km for fibre. In the experiment the sender node was Spain (ES) and the receiver node Norway (NO).

The traffic load parameters were the following: \( r_0 = 1\text{Mb/s} \), \( r_1 = 100\text{Mb/s} \), \( d_0 = 0.1\text{s} \), \( d_1 = 0.5\text{s} \), flow duration \( 1/\beta = 100\text{s} \) and total simulation time=86400s (24 hours). The simulation was repeated with an offered loads between 10Gb/s and 50Gb/s using different network loads (0%, 25%, 50%). The blocking ratio results are presented in Figure 5 and the results are very similar to the first experiment. This experiment was repeated using different sender and receiver nodes with no significant variation in results.

**B. Traffic distribution**

The previous experiments have evaluated the efficiency of the new scheme. Moreover, the efficiency of the PBC depends on the PDF of the traffic delay that transmits the channel. That is, the delay distribution must follow the distribution of figure 1. If the distribution has a long tail then, very few activations of the backup channel will be produced due to false failures.

Therefore, the following experiment simulates the delay distribution for a deadline of 0.2s using network of Figure 8. The network is loaded with 3 values (50%, 75% and 100%). Figure 10 represents the packet density function for different network loads: all packets arrive much sooner than their nominal deadlines and the more loaded the network is, the more delayed the packets are. For full load, the maximum packet delay is about 0.036s far beyond the nominal deadline (0.2s).

These results show that if we choose \( d_f = 0.1\text{s} \) (\( d_{\text{total}}/2 \)) there are practically no false failures, so the PBC scheme is very efficient. One question that arises is why the packet delays are so much lower than their deadlines. The main reason is the bursty characteristics of the traffic.

**V. PBC IMPLEMENTATION**

The PBC Scheme can be easily implemented as a part of a transport protocol on top of a network protocol. Nevertheless in this section we will detail it implementation using TCP trunking [27].
A TCP trunk is an aggregate traffic stream on layer-2 circuit or an MPLS path. Therefore, a TCP trunk has a known path and therefore an end-to-end delay. The routing algorithm can be executed in the ingress nodes or in a centralized node using MPLS.

The Proactive Backup Scheme uses an end-to-end failure detection scheme (see figure 11). The TCP trunk receiver node has the responsibility to detect if the channel fails and to begin the activation of the backup channel. The TCP trunk sender node injects a heartbeat into the channel when the sender is idle. The FSM (Failure Suspect Module) will provide the mechanisms to select the primary and backup trunks, activate the backup trunk when a packet delays more than $d_f$ and inject heartbeats in the network when there is not traffic. In absence of failures the backup trunk can transmit non-priority traffic.

Figure 12 shows a sample of how this method works. The sender sends a first packet with time $t_1$ and the time of the following packet $t_{prox}=t_2$. When this packet reaches the receiver, it knows that the next packet will arrive later than $t_{prox}+d_f$. For example, if packet 3 is lost due to a primary path failure, then the FSM assume a failure when time is $t_3+d_f$ and starts the activation of the backup trunk sending a NACK message to the sender (using the backup trunk). When the NACK message reaches the sender, it starts to transmit through the backup trunk keeping the transmission by the primary (the backup trunk resources are always reserved, so the backup trunk is ready to transmit). If the fail is confirmed (the receiver sends an NACK-C message) the sender will stop sending by the primary trunk.

If the fail is not confirmed (a false failure), the receiver will send an ABORT message to the sender to stop the transmission by the backup trunk. Therefore, in the case of a false failure this would imply the transmission of only two or three packets. In order this scheme to work correctly the network must provide a bounded and short set-up delay $d_s$. The path for this NACK message will be the backup trunk (it assumed to be OK). We stated in section 2 that the set-up time for the backup channel ($d_s$) is insignificant compared with the total delay ($d_{total}$). In the PBC implementation proposed in this section the set-up time $d_s$ is the time needed for a high priority message to reach the sender. In simulations this time is very low (about 187µs for IP networks [24]) so the assumption for $d_s=0$ is acceptable.

VI. CONCLUSIONS

This paper presents the application of the Proactive Backup Channel scheme for providing dependable channels with bounded delay. The idea behind this scheme is that packet delays are far behind its analytical bounds, so it is not necessary to wait for the primary path deadline in order to activate the secondary path. This is known as a proactive activation versus the traditional reactive activation.

The PBC scheme has been compared with the established Reactive Backup Channel (RBC) scheme. The experiments show that PBC can admit more dependable channels than RBC. This increment ranges from 10% to 40% depending on network load and topology.
This scheme is easy to implement using current network infrastructure. We detail a direct implementation using TCP trunking that shows the application of this scheme to current IP networks.

REFERENCES


