Budapest University of Technology and Economics
Department of Telecommunications and Telematics

Design and Performance Analysis of Routing Algorithms in Data Networks

Balázs Szviatovszki

Ph.D. Dissertation Summary

Advisors:

Dr. András Faragó
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas

Dr. Gyula Csopaki
Department of Telecommunications and Telematics
Budapest University of Technology and Economics

Dr. Gyula Sallai
Department of Telecommunications and Telematics
Budapest University of Technology and Economics

Budapest, Hungary
2002

1 Introduction

Looking back only twenty years, traffic was carried on dedicated networks. At that time the dominant traffic was generated by the telephony application that uses the circuit-switching principle. Deployment of converged multi-service networks only started recently as Internet and mobile traffic approached the volume of telephony traffic. Because of the strict quality of service (QoS) requirements, multi-service networks are based on the Asynchronous Transfer Mode (ATM) technology. However, research and standardization of QoS capable networks based on the packet-switching principle is currently the hottest topic nowadays, both in the telecommunication and Internet domains. Building a QoS capable multi-service network based on the Internet Protocol (IP) while at the same time keeping simplicity the guiding principle of development is not an easy task at all.

The framework for Traffic Engineering (TE) [1] defined by the Internet Engineering Task Force (IETF) provides a general framework for routing in IP networks. However, because of the new components added to the best-effort IP model there are many open issues concerning both equipment design and network engineering. The most important new components are:

- traffic engineering extension to link-state routing protocols;
- bandwidth constrained path computation algorithms;
- signalling protocols to establish paths in the network;
- traffic prioritization and preemption methods;
- restoration path configuration and path computation;

MultiProtocol Label Switching (MPLS) enables *explicit routing* of aggregated traffic flows via label switched paths (LSP), a similar concept to soft permanent virtual connection (SPVC) routing of ATM networks using the PNNI (Private Network-to-Network Interface) protocol [5]. An LSP reserves bandwidth on its path. When there is not enough capacity on the topological shortest path between the ingress and egress points of an LSP, a longer explicit path can be chosen to guarantee the required bandwidth. In both MPLS and PNNI a link-state routing protocol distributes topology and bandwidth reservation information of the network links. In MPLS this is achieved by flooding link-state advertisement (LSAs) with traffic engineering extension to e.g., the OSPF (open shortest path first) [4] protocol. In PNNI the equivalent of an LSA is called PNNI topology state element or PTSEs.

The topology and reservation information distributed in the network, enables devices to use bandwidth constrained algorithms during the path selection decision. *Constrained shortest path first (CSPF)* algorithms [8] are proposed by IETF to enhance best-effort IP routing. CSPF algorithms use flooded resource reservation information to construct a capacitated graph, on which such links can be neglected that e.g., do not have enough free bandwidth for the new LSP. On the resulted graph — depending on the requirements — path computation algorithms may range from simple shortest path routing using Dijkstra's algorithm, to more complicated ones using multiple link metrics. To fulfill operational requirements in many cases a path is searched on which not only the specified amount of bandwidth can be reserved, but which produces optimal or near optimal network resource utilization from a global perspective. Distributed algorithms of course can not achieve globally optimal resource utilization of the network, therefore, some kind of heuristic solution is searched. Typically, in these heuristic methods some kind of intelligent load-balancing technique is used [8]. When the path set of a network is planned in a central place, global path optimization is possible. There are a number of proposals in the literature to do this using Traffic Engineering tools [11], [J6].

LSPs are established on the computed explicit path with the help of extended versions of the RSVP [2] and LDP [3] protocols that carry so called Explicit Route Objects (EROs) to describe the strict hops to be followed.

At both path computation and path establishment, traffic prioritization plays an important role. By setting a high priority to delay-sensitive traffic it is possible to ensure that minimum hop paths are used during path computation. This is achieved by differentiating bandwidth reservations: the path computation algorithm of a given traffic priority is only informed about the reservations of similarly important traffic. Lower priority reservations are not taken into account, thus when performing path establishment preemption procedures are used to force e.g., best-effort traffic to find such paths that are not used by more important applications [10].

For certain applications the required bandwidth should be guaranteed irrespective of link failures. MPLS protection and restoration methods usually assume single link failures. One can calculate a pair of disjoint paths for which the *sum of the weights is minimal* using minimum cost flow algorithms [15]. One of the paths, typically the shorter one, is established as the working path, while the other one is stored as a backup path. When a link of a working path fails, the backup path is established to carry the protected traffic. After the failed link is restored, traffic may be reverted back to the original path. If the backup path is configured as non-revertive, the backup and the working path change roles, i.e., traffic is not reverted back to the original working path after the failure is cleared.

2 Research Objectives

The motivation of my dissertation is to study performance optimization problems in backbone data networks and to propose extensions, or alternatives to existing routing algorithms.

Throughout the dissertation, I will concentrate on routing issues arising in the network of a single operator, and I primarily assume that the link-state principle is used in the underlying interior routing protocol. In case of IP based networks, this means either the ISIS (intermediate system to intermediate system) or the OSPF (open shortest path first) protocols, while in ATM based networks I restrict my investigations to the PNNI (private network-to-network interface) protocol. My main interest is to study routing algorithms implemented within the path computation components of these protocols.

There are basically two possibilities for path calculation: local and global strategies. The former achieves greater automation by implementing the constrained shortest path first (CSPF) path selection algorithm in edge routers, while the latter performs a network-wide optimization of resource usage with the help of multi-commodity flow based algorithms.

I have defined the following as the main objectives of this dissertation:

- develop and analyze routing algorithms that increase the success probability of bandwidth constrained path establishment;
- develop and analyze backup path calculation algorithms taking into account operator requirements, such as fast restoration and optimal utilization of network links;
- investigate the effect of preemption in MPLS networks and the impact of routing schemes on the re-routing frequency caused by path preemption. Develop and analyze routing algorithms that avoid unnecessary preemption;
- investigate test methods for the performance evaluation of path computation implementations.

3 Methodology

The three basic performance evaluation methods of networking architectures are:

- analysis,
- simulation,
- prototyping and measurement.

Analytical methods for routing algorithms were extensively used for PSTN networks in the 80s. This approach provides the deepest insight into the behavior of a specific routing algorithm and most general results as well. As reviewed by Ash in [13], studies and results of many researchers (including Gibbens, Mitra and Kelly [12]) provided theoretical background for dimensioning of telephone networks with dynamic routing algorithms e.g., dynamic nonhierarchical routing (DNHR) or dynamic alternative routing (DAR). An important property of PSTN networks are that they are highly connected, all nodes are connected by a direct link, and only 2 hops long alternative paths are allowed for call routing. In general there are some properties of telephone networks that differ from those of contemporary data networks. The network topology in today's data networks has became sparser, network diameter has increased, and the traffic is hardly predictable (unlike in PSTN), therefore analytical approaches are barely usable in most cases.

For the performance evaluation of routing algorithms *simulation* techniques are used most frequently. This is the basic method of performance analysis I have used in my thesis. In simulations, the abstraction level can be set freely according to the needs of the problem instance. For most of the routing problems studied in this dissertation, a simplified routing model was appropriate which eliminated the need for detailed modelling of all link-state routing protocol components. Network flow problems often call for the use of *linear programming.* In Thesis 1 I use this method, as well.

Prototyping and measurements require the biggest development work, but provide the most accurate results. In Thesis 4. I propose an emulator architecture to evaluate real routing protocol implementations. The proposal was implemented in a prototype tool, with the help of which I have carried out performance evaluations of real equipment by using *measurements.*

4 New Results

Thesis 1 : LSP routing in an MPLS network by re-routing a single established LSP

The bandwidth constrained algorithm used for routing an MPLS label switched path (LSP) sometimes does not find a feasible path because there is no path between the ingress and egress routers on which the required bandwidth is available. In this case the straightforward action network operators can take is either to block the new LSP or to re-optimize the routing of the entire network to free some resources. The first case means lost revenue, because of lower throughput, the latter means major restructuring of LSP paths in the network.

Thesis 1.1 : An efficient algorithm for the LSP re-routing problem

I have proposed a new concept that eliminates the need for major network restructuring while increasing the setup probability of Label Switched Paths when the basic constrained shortest path first (CSPF) algorithm fails to find a satisfactory path. Aiming to minimize the required change in the network, I have proposed a new algorithm that *re-routes a single LSP* in order to establish a new LSP in an MPLS network [C5]. I concentrated on investigating the re-routability of a single LSP in order to minimize the running time of the algorithm. Although re-routing more LSPs might increase the success probability of the algorithm it probably does not justify the price paid in increased complexity.

The proposed algorithm consists of two main parts. The aim of the *first 'filtering' part* is to identify candidate re-routable LSPs, while the *second 'two-path' part* uses an integer linear programming based solution to decide whether the selected LSP and the new LSP can be established at the same time. The solution of the ILP problem may be time consuming, therefore the filtering part plays an important role in *decreasing the running time* of the algorithm. With the help of simple 'filters' it identifies a significant number of LSPs for which it is not worthwhile to execute the 'two-path' part.

In Fig. 1, the flowchart of the optimization algorithm is shown. If the `two-path` function finds a solution (i.e., there is a path for both the established LSPs and the new LSP), the optimization is complete, and the whole algorithm terminates. If there were no re-routable old LSPs, the algorithm terminates with failure.

Filtering

LSP re-routing trials that do not help the routing of the new LSP should be avoided. Consequently, the *first part* of the algorithm decreases the number of LSPs that have to be examined by the second ILP based function. For this purpose I have proposed three filters:

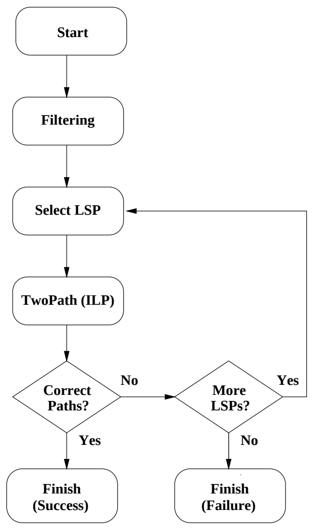


Figure 1: Flowchart of the optimization algorithm

1. Examine *bottleneck links* that do not have enough available bandwidth for the new LSP. First, the filter calculates the minimal number of bottlenecks for the new LSP by Dijkstra's shortest path algorithm with the following weight function:

$$w(e) = \begin{cases} 1 & \text{if edge } e \text{ does not have enough available bandwidth} \\ & \text{for the new demand,} \\ 0 & \text{otherwise.} \end{cases}$$

   Filter out such LSPs that contain less bottleneck links than the weight of the shortest path, using the weight function above. On at least one link of the paths of these LSPs there will not be enough bandwidth available for the new LSP, thus the evaluation of their re-routability is useless.

2. Those bottleneck links that are reachable from the new LSP's ingress node directly (i.e. without passing another bottleneck link), are called *first bottleneck links.* The LSP to be rerouted has to use one of the first bottleneck links. To find the first bottleneck links I have proposed the following algorithm:

   (a) Search for a shortest path between the LSP's ingress and egress router with the same weight function as above.

   (b) Search for the first bottleneck link of this path, delete it from the graph temporarily, and store it into a list.

   (c) Repeat the first two steps until there is no path between the ingress and egress nodes of the LSP demand.

When all possible first bottleneck links are found, established LSPs are filtered that do not use any of the bottleneck links of the list.

3. Check whether the unreserved bandwidth of the bottleneck link will be large enough to accommodate the new LSP after releasing the selected old LSP. This filter precludes the possibility of re-routing those established LSPs which do not facilitate the placement of the new LSP into the network, even if they are completely released.

Since each filter decreases the number of LSPs that should be examined by the `two-path` algorithm, the order in which the filters are executed can be chosen freely.

Two-path

To be able to solve the LSP re-routing problem, I have identified a sub-task called `two-path` . First the edges of the graph are used to fill up three sets according to the following rule: edges in sets E_1 and E_2 have enough capacity to accommodate the first path and the second path, respectively (the two path may have different capacities). Edges in E_b can not accommodate the two paths together. The task of `two-path` is to find two paths P_1 and P_2 from s_1 to t_1 and from s_2 to t_2 such that $P_1 \subseteq E_1$, $P_2 \subseteq E_2$ and $P_1 \cap P_2 \cap E_b = 0$. In other words, the path P_i can use only the edges of E_i and furthermore, the edges which are used by both paths must not be in E_b.

It can be shown that even this basic problem is NP-hard. I have proposed a method which is based on the following *integer linear programming* formulation:

I seek the binary vectors x_1 and x_2, satisfying the conditions:

$$x_1^e \in \{0,1\} \qquad \forall e \in E_1 \tag{1a}$$

$$x_2^e \in \{0,1\} \qquad \forall e \in E_2 \tag{1b}$$

$$\sum_{e \in \rho(v) \cap E_1} x_1^e - \sum_{e \in \delta(v) \cap E_1} x_1^e = \delta_{v,t_1} - \delta_{v,s_1} \qquad \forall v \in V \tag{1c}$$

$$\sum_{e \in \rho(v) \cap E_2} x_2^e - \sum_{e \in \delta(v) \cap E_2} x_2^e = \delta_{v,t_2} - \delta_{v,s_2} \qquad \forall v \in V \tag{1d}$$

$$x_1^e + x_2^e \leq 1 \qquad \forall e \in E_b, \tag{1e}$$

and approach

$$\min \sum_{e \in E_1} c_1(e) x_1^e + \sum_{e \in E_2} c_2(e) x_2^e \tag{2}$$

where $\rho(v)$ and $\delta(v)$ are the sets of the incoming and the outgoing edges of the node v and $\delta_{i,j}$ is 1 or 0 according to whether i is equal to j or not. The two cost functions $c_1, c_2 : E \longrightarrow \mathbf{R}_+$ are assigned to the edges. Variable x_i^e indicates whether path i uses edge e or not.

To efficiently solve the formulated ILP problem any ILP software package can be used. I have tested the `lp_solve` package [17] and it gave good results, even though the problem is NP-hard. The `lp_solve` package uses the *simplex method* to solve the LP problem.

Thesis 1.2 : Performance analysis of the proposed LSP re-routing algorithm

I have conducted simulation experiments to evaluate the performance of the proposed LSP routing algorithm [C5]. For the simulation I used the AT&T IP Backbone network [18], that consists of 25 backbone nodes and 88 links, resulting in an average node degree of 3.52. I created randomly generated traffic situations and for given traffic situations (characterized by the *total throughput*, i.e., the sum of all established LSPs' bandwidth) I compared the proposed algorithm to the basic CSPF algorithm that blocks the LSP setup when there are not sufficient resources available in the network.

I have concluded that the proposed algorithm:

- largely improves the success probability of new LSP establishments (Fig. 2);
- scales very well for practical network sizes, even though the problem is NP-hard. The running time of the algorithm was 4.6 seconds for a 150 node network and 8.2 seconds for a 200 node network on a Sun UltraSparc 5 workstation using the `lp_solve` package.

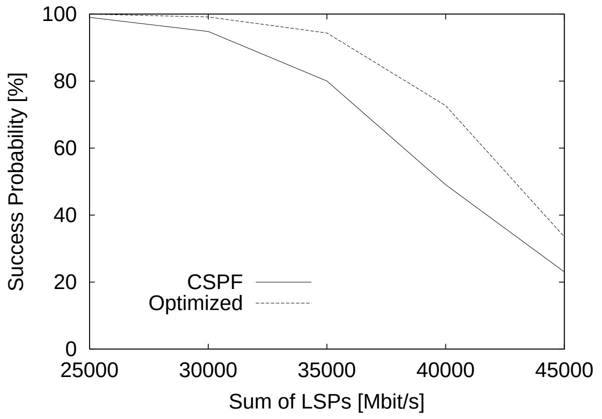


Figure 2: Success probability of the new algorithm, same initial states

I have found that the proposed filtering functions decreased the running time of the algorithm significantly. I have shown that the running time of the algorithm can be further decreased by solving the ILP problem with the *Column Generation Procedure* [16] instead of the simplex method (used by the `lp_solve` package).

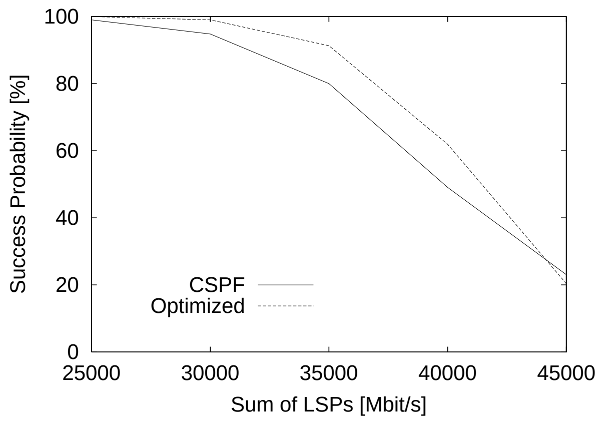


Figure 3: Success probability of the new algorithm, different initial states

In the previous experiments the required network load was achieved by establishing LSPs with the simple CSPF method (Fig. 2). I have also verified the applicability of the algorithm by checking whether the performance improvement remains significant when the new LSP re-routing based method is used for the establishment of all LSPs in the network from the very beginning of each simulation experiment. In this case, for all failed LSPs establishments a trial is made to re-route an already established LSP. This results in longer average path length and thus at a given carried traffic level higher average link loads. Due to this, I expected that using different initial states the success probability measures will be slightly lower compared to the same initial state scenario.

I have concluded that the proposed algorithm:

- results in significantly higher success probability than the simple CSPF method at reasonable network loads (Fig. 3);
- the performance difference decreases at higher average link loads (above 37000Mbit/s total carried traffic, representing 60% average link load) when re-routing resulted in longer average path length;

Moreover, I have conducted simulation experiments to evaluate how much performance improvement would be achieved if the algorithm was implemented in a distributed way in real router code. This application allows only the re-routing of those LSPs that originate from the ingress router of the new LSP. Simulation results showed that even in this case, the success probability of new LSP establishment can be increased by approximately 2-3%. However, this represents only a marginal gain and therefore the distributed implementation of the algorithm has less practical relevance.

Thesis 2 : Shortest path based restoration method

When designing a specific restoration architecture it is important to take into account operational requirements. One of the most important requirements for backbone network providers is to minimize service interruption time and achieve e.g., 99.99% service availability. This requires fast restoration methods that scale well when there are hundreds or thousands of connections to be restored after a single link failure event. As backbone networks should be kept very stable, it is of fundamental importance to operators to minimize the restructuring of the network and to simplify the reversion cycle after the failed link is taken back to service.

Authors in [19] point out that restoration path computation methods assuming pre-computation, but not pre-establishment of backup paths provide fast restoration and good resource utilization at the same time. To achieve optimal resource utilization, it is important to use paths that are as short as possible for routing a given connection. Although the problem of finding a shortest path having a shortest disjoint backup path is NP-hard, an *integer linear programming problem* can be formulated that provides good practical results, as reported by [20]. However, if we relax the constraint that only one restoration path should protect against the failure of working path, we get a new, yet unresolved problem area.

Thesis 2.1 : Shortest backup path computation algorithm

I have proposed a set of operational principles for a new MPLS backup path computation method [C8]:

1. Backup paths are pre-computed, but not pre-established;
2. The working path and the non-revertive backup path are both shortest paths;
3. The number of common edges in the working path and the non-revertive backup path should be minimized;
4. If the working path and the non-revertive backup path are disjoint, no more backup paths are needed;
5. If there are common edges in the working path and the non-revertive backup path, another path is computed that protects against the failure of the common links. This path will be used as a revertive backup path;

Among these requirements number 2 represents the basic novelty, number 5 is a consequence, while the others are needed to fully specify the environment.

An algorithm based on the above operational principles is novel, since it minimizes resource usage of backup paths, furthermore, it increases the probability that non-revertive backup paths can be configured. In commercial router implementations (e.g., in [21]) it is possible to store more backup paths for a working path and to configure them to be revertive or non-revertive. The rational behind pre-computation is that it allows faster restoration time than can be achieved with on-demand computation. The computation of the second non-revertive backup path may be further optimized, but it is believed that it has very limited effect on the most important measures, namely on the average secondary path length and the probability that the backup path is a shortest one. Therefore the only requirement I have formulated is to be disjoint from the common edges of the working and the non-revertive backup paths.

In order to provide algorithms that can compute paths satisfying the above requirements, I have identified two sub-tasks. One is to *compute two shortest paths having the minimum number of common edges* (sub-task 1, to fulfill requirement number 2 and 3), while the other is to *compute a third path that provides full protection for the working path* (sub-task 2, to fulfill requirement number 5). The proposed algorithm for each sub-task is given below.

I have proposed the following algorithm for computing two shortest paths having minimum number of common edges (sub-task 1) [J1], [C8]:

1. Determine the minimum path cost $\pi_{min}(v)$ for each node v in the graph by running Dijkstra's shortest path algorithm from a fixed node s;

2. Delete all uv edges $\in E$ for which $\pi_{min}(v) - \pi_{min}(u) \neq w(u, v)$;

3. Duplicate every (remaining) edge, and assign weight '0' for one edge, and weight '1' for the other;

4. For the modified graph, determine two paths P_1, P_2 with minimum total path cost, by using a two-valued unit capacity minimum cost flow algorithm (sometimes referred to as Suurballe's method).

The two paths obtained by the minimum cost flow algorithm will be two shortest paths in the original graph, having a minimum number of common edges. If the two obtained paths are totally disjoint, one of them will be used as the working path, the other as a non-revertive backup path. In this case there is no need for another backup path. In case of identical paths, one is kept as a working path, the other one is deleted.

The third path guaranteeing protection for the failure of any link of the working path is computed with the following algorithm (sub-task 2) [C8]:

1. Start with the original graph;

2. Assign a weight $w(e)$ for the common edges of the first two shortest paths $\mid P_1 \cap P_2 \mid$ that is larger then the summarized weights of all edges in the graph. This means that all the edge weights of the working path will be increased if the two paths were identical in the previous step;

3. Determine a minimum cost s, t path by running Dijkstra's shortest path algorithm;

Thesis 2.2 : Performance analysis of the proposed restoration method

I have conducted simulation studies to analyze the performance of the proposed restoration path computation methods [C8]. As a reference in the experiments I have used two algorithms discussed in [14], namely the 'Suurballe' method and the 'Penalty' method.

In the first experiment to characterize the network itself, I have determined basic measures for the Cable & Wireless and the AT&T IP network topologies [18]. I have concluded that:

- the proposed algorithm found two shortest paths for 35-40% of the node pairs in the network, while traditional algorithms achieved only 15-20%; moreover
- more than half of the obtained paths have only one common edge, meaning that it is highly probable that a shortest path can be used for restoration. This is achieved with a non-revertive backup path.

In another experiment I used the AT&T network and random traffic to measure several performance factors of different methods. I have concluded that:

- the proposed backup path computation algorithm's average backup path length measure is approximately 8-9% less than the average path length of the two reference algorithms (consistently for all investigated network load levels).
- the probability that a shortest backup path can be used for restoration is doubled with the proposed algorithm (Fig. 4). This means that after a failure when LSPs are re-routed, network resources will be used more efficiently and moreover, these LSPs do not necessarily have to be reverted back to their original path after the failure is restored.

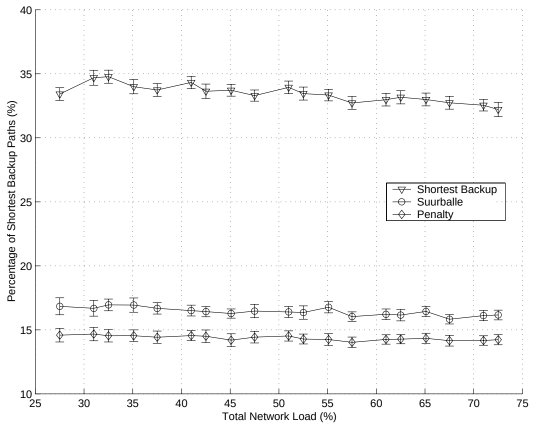


Figure 4: Probability of finding a shortest backup path

In the last experiments I have checked the validity of the previous results on random networks having a random number of nodes and link densities. I have found that as the number of nodes in the network increases, so does the performance of the ‘Shortest Backup’ method. Similarly if the number of nodes are fixed and the edge densities are increased, the difference between the algorithms become more noticeable.

Thesis 3 : Preemption-aware constraint-based routing in MPLS networks

In order to compute explicit paths in MPLS networks, bandwidth constrained shortest path first (CSPF) algorithms need to know the actual load levels on the links. Traffic Engineering extensions to link-state interior routing protocols distribute the maximum reservable bandwidth, B_{max}, and the unreserved bandwidth of each link in the network, $\mathbf{B}_u = (B_{u0}, B_{u1}, B_{u2}, \cdots, B_{u7})$. There are eight B_u values, since in MPLS eight priority levels are defined to support service differentiation. To make path selection bandwidth constrained, the CSPF algorithm in the edge router prunes non-conforming links based on the bandwidth check $B_{us} < B_{LSP}$, where B_{LSP} is the required bandwidth and s is the setup priority of the LSP. When doing path computation on the remaining graph, all known CSPF algorithms take into account only the unreserved bandwidth values of the new LSP's setup priority level, but neglect any lower level unreserved bandwidth information. Although at path setup LSPs with lower priority can be preempted, known CSPF algorithms do not take this into account at path computation.

Thesis 3.1 : Preemption measures

I have proposed to take into account unreserved bandwidth information of lower priority levels when computing the path of an LSP [J4], [C7]. For this purpose, I have identified two preemption measures that can be derived from flooded link-state information, namely:

1. *free bandwidth*, i.e., the unreserved bandwidth at the lowest (7^{th}) priority level:

$$B_{free} = B_{u7} \tag{3}$$

2. *bandwidth preemption vector*:

$$\mathbf{B}_p = (B_{p0}, B_{p1}, B_{p2}, \cdots, B_{p7}) \tag{4}$$

   which is a complex measure containing the vector of estimated bandwidth values that should be preempted at each priority level of a link. With the help of this measure we can determine the *highest affected priority level* on a link, i.e., the smallest such number i for which $B_{pi} \neq 0$. In Fig. 5 I propose a procedure to calculate the bandwidth preemption vector.

```
procedure CalcBwPreemptionVector($\mathbf{B}_u, B_{LSP}$)
    $\mathbf{B}_p = 0$
    $B_{u(-1)} = B_{max}$
    for $(i = 7, i \geq 0, i\text{--})$
        if $B_{LSP} \leq B_{ui}$ return $\mathbf{B}_p$
        $B_{pi} = min((B_{LSP} - B_{ui}), (B_{u(i-1)} - B_{ui}))$
    end for
    return $\mathbf{B}_p$
end procedure
```

Figure 5: Calculating the bandwidth preemption vector

Thesis 3.2 : Application of preemption measures in routing algorithms

I have proposed new preemption-aware CSPF algorithms by using the preemption measures defined in Thesis 3.1 [J4], [C7].

For the *free bandwidth* measure I propose to use the bottleneck link's free bandwidth measures to determine the metric of the path in Dijkstra's algorithm. When comparing the measure of two paths, I propose to choose the path with larger free bandwidth as a better candidate path.

When accumulating two links' measure, I propose that the larger $\mathbf{B}_p$ vector is taken as the path's metric. I propose to use lexicographical comparison for the bandwidth preemption vector ($\mathbf{B}_p$) measure of two paths.

I proposed for all preemption-aware algorithms to first prune links for which $B_{us} < B_{LSP}$. Then on the resulting graph I proposed to restrict path selection to shortest paths based on the original OSPF metric. I proposed to use preemption information when selecting a candidate path among otherwise shortest feasible ones. In this way the new metrics that minimize preemption do not adversely affect the CSPF success ratio and path length of high priority LSPs. Possible ways of constructing preemption-aware CSPF algorithms are summarized in Table 1.

Table 1: Metric ordering in the proposed algorithms

	1^{st} metric	2^{nd} metric	3^{nd} metric
1	shortest	free bandwidth	—
2	shortest	bandwidth preemption vector	—
3	shortest	bandwidth preemption vector	free bandwidth

Thesis 3.3 : Performance evaluation of the proposed preemption-aware constraint-based routing algorithms

I have conducted simulation studies to analyze the performance of the proposed preemption-aware CSPF algorithms [J4], [C7]. The first simulated network was the *Cable & Wireless* backbone network topology [18] that contains 31 backbone nodes and 102 links, resulting in an average node degree of 3.29. I created randomly generated traffic situations. This means that the network was loaded to a certain extent with randomly placed LSPs having random bandwidth values. LSP bandwidth was uniformly-distributed within the interval $(0, B_{MaxLSP}]$. In the experiments I focused on such situations when B_{MaxLSP} was around 8-10% of the most common OC-12 link capacity which is in line with the assumptions of Rexford [7]. The priority levels of the LSPs were also set randomly with uniform-distribution in [0-7].

I have selected different load-balancing algorithms from the literature for reference (namely the 'widest-shortest path' method [6], the 'discrete link cost' method [7] and the 'residual bandwidth ratio' method [8]).

I have concluded that:

- the proposed preemption-aware CSPF algorithms are very close to the best CSPF algorithms in terms of success probability (within 1-2%) irrespective of average network load;
- the probability of preemption is significantly lower for the proposed preemption minimization methods. The strategy to minimize the affected priority levels is more effective than the simpler one that maximizes the bottleneck free bandwidth of the path. At high loads the former achieves 10%, while the latter 5% improvement compared to other load-balancing methods;
- the number of LSPs in the preemption chain does not differ significantly for different methods;

In order to provide more general results, I have carried out simulation experiments on random networks as well, with a varying number of nodes and average node degree. The experiments show similar results to the real-world network simulations. The achieved improvement was in the 3-11% range. I have noticed that as the number of nodes and the average node degree increases, so does the improvement that we can achieve by using preemption-aware CSPF techniques.

Thesis 4 : Architecture for testing real path selection algorithm implementations

Simulation of routing algorithms provides an important means for the performance evaluation of backbone data networks. When a specific algorithm is selected and implemented in routing software modules, it is of prime interest of manufacturers to thoroughly test the new implementation. Therefore, beside simulation studies, conformance and scalability tests should also be performed. For conformance tests, i.e., for checking whether the code implements functions according the specification, simple test networks are appropriate. Scalability tests, on the other hand, require larger test networks. However, there can be situations when there are no suitably large networks built, or because those built are already used for commercial services.

I have proposed an emulator architecture [C3] for testing the performance of real implementations of the PNNI (private network-to-network interface) protocol [5]. It enables an implementation under test (IUT) to be considered as part of the simulated network, thus the IUT is forced to function as if it were part of a real network. In this way, test configurations can be prepared by configuring the simulator (editing input files), rather than setting up real equipment with dozens of interconnections between them.

I have identified the following components of the architecture that are of vital importance:

- a simulator running on a workstation;
- the simulator should model the PNNI routing and signalling protocol in its details: the Hello, Database Synchronization and Flooding functions should be implemented according to the PNNI specifications [5];
- with the help of an efficient topology database implementation the simulator should use an abstraction of the simulated network state in order to enable efficient memory use and simplified implementation of protocol message handling in simulated nodes;
- the workstation running the simulator should be equipped with an ATM card for sending and receiving ATM AAL5 packets;
- interface functions should be implemented in the simulator to convert simulated messages to bit-conform PNNI signaling messages, and vica versa;
- the simulator has to operate in real-time mode and not in the simple event-driven mode;
- the architecture should support the connection of an IUT with more than one link (routing control channels) to the emulator;

In Fig. 6 the functional diagram of a two node network is shown: a real ATM switch is interconnected with the emulator modelling another switch.

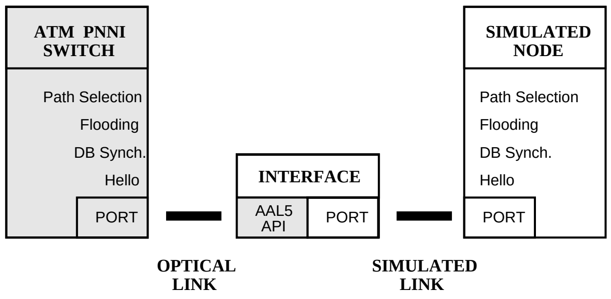
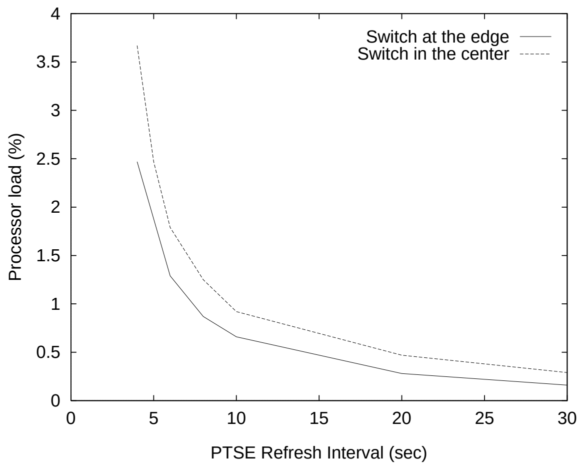


Figure 6: The functional diagram of the emulator modelling a two node network

Based on the proposed emulator architecture I have developed a prototype emulator implementation. The base for the implementation was an available simulator implementation [22]. With the help of a prototype implementation of the emulator architecture, I evaluated the performance of a commercial ATM switch. The aim of the experiment was to validate the applicability of the proposed architecture. The processor load was measured as a function of flooding intensity. Frequent flooding of link-state descriptor protocol packets was generated by changing the 'PTSE Refresh Interval' settings in the emulated nodes' configuration. Results in Fig. 7 were obtained for two network configurations; in one the IUT was connected to the edge of the network through one link, while in the other, two links were used, thus the IUT was tested as if it was in the middle of a real network. The findings on the networks do not represent fundamental scientific novelty (as this was not my goal), however, with the experiments I have shown that the emulator is applicable for testing PNNI protocol implementations of commercial ATM switches.



Settings

Real switch and Emulator:

MinPTSEInterval: 0.1s,

PTSELifetimeFactor: 101%,

PeerDelayedAckInterval: 1s.

Real switch:

PTSERefreshInterval: 30s,

Emulator:

PTSERefreshInterval:

decreased from 30s down to 4s.

Figure 7: The processor load of a real switch connected to the emulator

5 Application of the Results

The traffic engineering and routing optimization research was carried out as a part of projects partly sponsored by Ericsson Research and Ericsson Business Units. The new routing algorithms that are proposed and evaluated in my dissertation can support equipment development of Ericsson products in the future. Performance evaluation results of both new and known algorithms supports business units of Ericsson in selecting the best algorithms to be implemented in router software.

The preemption-aware bandwidth constrained routing algorithm presented in Thesis 3. has been patented.

The emulator architecture in Thesis 4 has been implemented in a prototype tool and tested using Cisco ATM Switches.

References

[1] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja and X. Xiao, *A Framework for Internet Traffic Engineering*, Internet Engineering Task Force, Internet Draft, April 2001, Work in Progress.

[2] D. Awduche, L. Berger, D.H. Gan, T. Li, G. Swallow, V. Srinivasan, *RSVP-TE: Extensions to RSVP for LSP Tunnels*, Internet Engineering Task Force, Request For Comments (Proposed Standard) 3209, December 2001.

[3] B. Jamoussi (editor), *Constraint-Based LSP Setup Using LDP*, Internet Engineering Task Force, Request For Comments (Proposed Standard) 3212, January 2002.

[4] D. Katz, D. Yeung and K. Kompella, *Traffic Engineering Extensions to OSPF*, Internet Engineering Task Force, Internet Draft, March 2001, Work in Progress.

[5] ATM Forum Technical Committee: *Private Network-Network Interface Specification Version 1.0*, ATM Forum af-pnni-0055.000, March 1996.

[6] G. Apostolopoulos, R. Williams, S. Kamat, R. Guerin, A. Orda, A. Przygienda, *QoS Routing Mechanisms and OSPF Extensions*, Internet Engineering Task Force, Request For Comments (Proposed Standard) 2676, August 1999.

[7] A. Shaikh, J. Rexford and K.G. Shin, *Evaluating the impact of stale link state on quality-of-service routing*, IEEE/ACM Transactions on Networking, vol.9, no.2, p.162-176, April 2001.

[8] K. Kompella and D. Awduche, *Notes on Path Computation in Constraint-Based Routing*, Internet Engineering Task Force, Internet Draft, September 2000, Work in Progress.

[9] V. Sharma et al, *Framework for MPLS-based Recovery*, Internet Engineering Task Force, Internet Draft, March 2001, Work in Progress.

[10] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus, *Requirements for Traffic Engineering Over MPLS*, Internet Engineering Task Force, Request For Comments (Proposed Standard) 2702, September 1999.

[11] P. Aukia, M. Kodialam, P.V. Koppol, T.V. Lakshman, H. Sarin, B. Suter, *RATES: A server for MPLS traffic engineering*, IEEE Network, vol.14, no.2, pp.34–41, 2000.

[12] F. P. Kelly, *Bounds on the performance of dynamic routing schemes for highly connected networks*, Mathematics of Operations Research, Vol.19, No.1, February 1994.

[13] G. R. Ash, *Dynamic Routing in Telecommunications Networks*, chapter 8.2 "Real-Time Dynamic Routing Methods", pp 299–308, ISBN 0-07-006414-8, McGraw-Hill, 1997.

[14] A. Iwata, R. Izmailov, B. Sengupta, *Alternative Routing Methods for PNNI Networks with Partially Disjoint Paths*, IEEE Globecom 98, November 1998.

[15] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, Upper Saddle River, New Jersey 07458. 1993.

[16] A. Schrijver, *Theory of Linear and Integer Programming*, Jowh Wiley&Sons, Wiley-Interscience series in discrete mathematics, 1986.

[17] M. Berkelaar, J. Dirks, *lp_solve 2.2*,
ftp://ftp.es.ele.tue.nl/pub/lp_solve

[18] Russ Haynal's ISP Page,
http://navigators.com/isp.html

[19] R. Doverspike, J. Yates, *Challenges for MPLS in Optical Network Restoration*, IEEE Communications Magazine, Vol. 39, No. 2, pp.89-96, February 2001.

[20] P. Laborczi, J. Tapolcai, P.-H. Ho, T. Cinkler, A. Recski and H.T. Mouftah, *Algorithms for Asymmetrically Weighted Pair of Disjoint Paths in Survivable Networks*, Third International Workshop on Design of Reliable Communications Networks, (DRCN) October 2001.

[21] A. Statti, P, Muraca, C. Morris, *JUNOS 5.0 Internet Software Configuration Guide:MPLS Applications*, Juniper Networks, Inc. 1194 North Mathila Avenue Sunnyvale, CA 94089 USA, August 2001
http://www.juniper.net/techpubs

[22] Á. Szentesi, *Routing and Topology Optimization in ATM Networks*, Ph.D thesis, Department of Telecommunication and Telematics, Budapest University of Technology and Economics, 2002.

Publications

Journal papers

[J1] Á. Magi, Á. Szentesi, B. Szviatovszki, A. Faragó: *Dinamikus útvonalválasztás ATM hálózatokban*, Híradástechnika, Vol. 50, No. 11, pp. 2–11, November 1999.

[J2] Á. Magi, Á. Szentesi, B. Szviatovszki: *Analysis of link cost functions for PNNI routing*, Computer Networks, Vol. 34, No. 1, pp. 181–197, July 2000.

[J3] A. Faragó, Á. Szentesi, B. Szviatovszki: *Inverse Optimization in High Speed Networks*, To appear in the Spec. Issue on Combinatorial and Algorithmic Aspects of Telecom., Discrete Applied Mathematics Journal (submitted: January 1998, reviewed: April 1999)

[J4] B. Szviatovszki, Á. Szentesi, A. Jüttner: *Minimizing Re-Routing in MPLS Networks with Preemption-Aware Constraint-Based Routing*, Computer Communications Journal, Spec. Issue on Advances in Performance Evaluation of Computer and Telecommunications Networking, Vol. 25, No. 11-12, pp. 1076-1084, May 2002.

[J5] D. Orincsay, B. Szviatovszki, G. Böhm: *Prompt Partial Path Optimization in MPLS networks*, Submitted to The International Journal of Computer and Telecommunications Networking, (submitted: November 2002)

[J6] J. Forslow, I. Jarrett, P. Moran and B. Szviatovszki: *Management solutions for IP networks*, Ericsson Review. vol.77, no.1, p.42-52. 2000

Conference and workshop papers

[C1] A. Faragó, Á. Szentesi, B. Szviatovszki: *Allocation of Administrative Weights in PNNI*, In proc. of the 8th International Telecommunication Network Planning Symposium, Networks'98, pp. 621–626, Sorrento, Italy, October 18–23, 1998

[C2] Á. Magi, G. Nagy, Á. Szentesi, B. Szviatovszki: *Analysis of link cost functions for PNNI routing*, In proc. of the 6th IFIP Workshop on Performance Modeling and Evaluation of ATM Networks, pp. 50/1–50/10, Ilkley, UK, July 20–22, 1998

[C3] Á. Magi, B. Szviatovszki, G. Rózsa, L. Németh: *Novel Performance Evaluation of PNNI Equipment*, In proc. of the XVII World Telecommunications Congress incorporating ISS'2000, Birmingham, UK, May 7–12, 2000.

[C4] B. G. Józsa, D. Orincsay, Á. Magi, B. Szviatovszki: *On the Use of Trunk Reservation in PNNI Routing*, In proc. of the IEEE Conference on High Performance

Switching and Routing (ATM2000), pp. 135–139, Heidelberg, Germany, June 26–29, 2000.

[C5] A. Jüttner, B. Szviatovszki, Á. Szentesi, D. Orincsay, J. Harmatos: *On-demand optimization of label switched paths in MPLS networks*, In proc. of the 9th International Conference on Computer Communications and Networks, IEEE ICCCN 2000, pp. 107–113, Las Vegas, USA, October 16–18, 2000.

[C6] A. Jüttner, B. Szviatovszki, I. Mécs, Zs. Rajkó: *Lagrange Relaxation Based Method for the QoS Routing Problem*, In proc. of the IEEE Infocom 2001, Anchorage, Alaska, USA, April 22–26, 2001.

[C7] B. Szviatovszki, Á. Szentesi, A. Jüttner: *Minimizing Re-Routing in MPLS Networks with Preemption-Aware Constraint-Based Routing*, In proc. of the 2001 International Symposium of Performance Evaluation of Computer and Telecommunication Systems, pp. 249–261, Spects 2001, Orlando Florida, July 15–19, 2001

[C8] B. Szviatovszki, Á. Szentesi, A. Jüttner: *On the effectiveness of Restoration Path Computation Methods*, In proc. of the IEEE International Conference on Communications, New York, April 28–May 2, 2002.

Patents

[P1] B. Szviatovszki, Á. Szentesi, A. Jüttner: *Priority-Based Path Selection in a Data Network Control*, (US Patent Application No. P14200US, 2001)