Network Dimensioning for the Self-Protecting Multipath: A Performance Study

Michael Menth, Ruediger Martin, and Ulrich Spoerlein
Department of Distributed Systems, Institute of Computer Science, University of Würzburg, Germany
Email: {menth,martin,spoerlein}@informatik.uni-wuerzburg.de

***Abstract*— The self-protecting multipath (SPM) is a simple protection switching mechanism that can be implemented, e.g., by MPLS. We present a linear program to optimize the SPM load balancing parameters for network dimensioning. Our study shows that the SPM is a very efficient mechanism in the sense that it requires only little backup capacity since it outperforms the p-cycle approach and the shortest path rerouting by far. The investigation of the computation time and the memory consumption recommends the Simplex method as an LP solver rather than an interior point method (IPM). The computation time of the program depends mainly on the number of links in the network and it is well feasible for small and and medium size networks. For large networks, however, fast heuristics are required.**

I. Introduction

Carrier grade networks require high availability which is often as high as 99.999% such that restoration or protection switching is required. Restoration sets up a new path after a failure while protection switching pre-establishes backup paths in advance. A typical restoration scheme is routing convergence in IP networks, which heals broken paths some time after a failure. A typical protection switching mechanism is the primary and backup path concept, where the traffic is switched onto the backup path as soon as the primary path does not work anymore. Protection switching or restoration mechanisms alone are not sufficient to maintain the full service availability during network failures. Then, the links carry the normal traffic together with the traffic that is deviated by the resilience mechanism. As a consequence, the quality of service (QoS) can only be met if the links have enough capacity. This must be taken into account for network provisioning. If the link capacities are already given, the structure of the backup paths must be laid out in such a way that they have enough capacity for all relevant failure scenarios.

In this paper, we focus on the self-protecting multipath (SPM) which is a protection switching mechanism that has been proposed in previous work [1], [2]. The SPM consists of several parallel paths between source and destination, and a load balancing function distributes the traffic over the working paths. The particularity of that concept is that the traffic may be spread over several paths both under normal networking conditions and in case of network failures. First, a multipath structure for the SPM is found and then, the load balancing function can be optimized. The contribution of this paper is a concise presentation of a linear program (LP) that optimizes the load balancing function of the SPM for network dimensioning in such a way that the overall required network capacity is minimized, which is needed to carry and protect the traffic. In addition, the complexity of the LP is investigated both theoretically and by empirical data. This is crucial for the assessment of the practical applicability of this optimization approach.

This paper is organized as follows. Section II gives an overview on protection switching techniques. Section III explains the LP for the optimization of the SPM load balancing functions and analyzes its complexity. Section IV investigates the required backup capacity of the SPM; furthermore, computation time and memory consumption of the optimization program are studied by experimental data. Finally, the conclusion in Section V summarizes this work and gives an outlook on further research.

II. Overview on Resilience Mechanisms

In this section we give a short overview on various resilience mechanisms to contrast the SPM against other approaches.

A. Restoration Mechanisms

As mentioned before, restoration mechanisms take actions only after a network failure. They try to find new routes or set up explicit backup paths when the traffic cannot be forwarded anymore due to link or node failures. The disadvantage of such methods is obvious: they are slow. The re-convergence of the IP routing algorithm is a very simple and robust restoration mechanism [3], [4]. Another example are backup paths in MPLS that are set up after a network failure.

B. Protection Switching Mechanisms

The authors of [5] give a good overview on different protection switching mechanisms for MPLS.

1) End-to-End Protection with Primary and Backup Paths: Backup paths are set up simultaneously with primary paths and in case of a failure, the traffic is just shifted at the path ingress router of a broken primary path to the corresponding backup path. This is called end-to-end protection. It is faster than restoration methods but the signalling of the failure to the path ingress router takes time and traffic already on the way is lost.

2) Fast Reroute Mechanisms: MPLS fast reroute (FRR) tackles the problem of lost traffic in case of end-to-end protection. Backup paths towards the destination are set up not only at the ingress router of the primary path but at almost every node of the path [6], [7]. Then, a backup path is immediately available if the path breaks at some location. Currently, fast reroute mechanisms are also discussed for IP networks. Several solutions are being discussed but a preferred method is not yet established [8]–[11].

3) p-Cycles: The protection cycle (p-cycle) approach [12], [13] is a protection switching mechanism originally designed for ring structures. "On-cycle paths" are protected by the complementary path on the ring in counter direction. "Straddling paths" cut the ring twice and can be protected by the both halves of the p-cycle operating in opposite direction. P-cycles are known as very efficient protection switching mechanism because they can be configured such that less than 50% backup capacity is required [14]. In Section IV-A.2 we compare the efficiency of the SPM and the one of p-cycles.

4) Self-Protecting Multipath: The self-protecting multipath (SPM) has been presented first in [1], [2]. Its path layout consists of disjoint paths and the traffic is distributed over all of them according to a traffic distribution function (see Figure 1). If a single path fails, the traffic is redistributed over the working paths according to another traffic distribution function such that no traffic is lost. Thus, a specific traffic distribution function is required for every pattern of working paths.

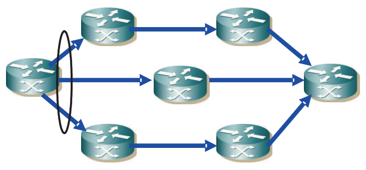


Fig. 1. The SPM performs load balancing over disjoint paths according to a traffic distribution function which depends on the working paths.

C. Routing Optimization

The traffic matrix and the paths of the flows together determine the resource demands on the links. The layout of the paths may be optimized to minimize either the link utilization or the required network capacity. In the following, we address briefly different optimization objectives to distinguish our optimization problem from others.

1) Routing Optimization for Networks with Given Link Capacities: In already provisioned networks, the capacity of the links is fixed. If the traffic matrix is given, the maximum link utilization in the network under failure-free conditions can be minimized by a suitable routing. This has been done for IP networks [15], for MPLS networks, and for hybrid networks [16]. If restoration or protection switching is applied, the target may be to minimize the maximum link utilization in any failure case. This has been done for IP networks [3], [4] and for MPLS networks [17]. Thereby, backup capacities may be shared by different flows and in different failure scenarios.

2) Routing Optimization in Combination with Network Dimensioning: In not yet provisioned networks, the network capacity and the routing may be determined together. If failure scenarios are not taken into account, shortest path routing requires the least capacity. With resilience requirements, however, backup resources may be shared by different flows in different failure scenarios. Routing optimization can reduce the required network capacity considerably by maximizing the capacity sharing. This has been exemplified by [18] and [1] and we also focus on this problem for the SPM in our work.

III. Optimization of the SPM for Capacity Dimensioning

The SPM consists of parallel paths over which the traffic is distributed according to a load balancing function. Thus, a suitable choice of the multipath layout and the optimization of the path failure specific load balancing function can minimize the required capacity to support a given traffic matrix. First, we describe a heuristic for the path layout, then we explain the linear program for the optimization of loaf balancing function, and finally, we analyze the complexity of the linear program.

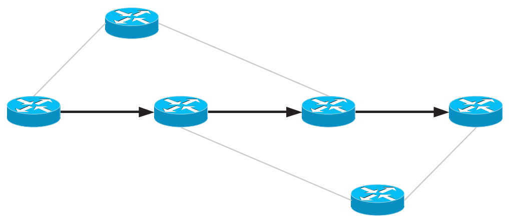
A. Path Layout

The SPM consists of disjoint parallel paths such that the remaining paths are still working if one path fails due to the failure of a single network element. In some cases the network topology does not allow to find disjoint paths. However, in this theoretical investigation, we do not consider this case, and in practice, there are workarounds to cope with that problem. We shortly mention two different algorithm families that help finding link or node disjoint paths in a network. Finally, we clarify the relation to shared risk link groups (SRLG).

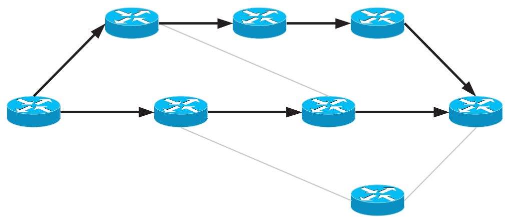
1) Iterative Application of the Shortest Path Algorithm: A very intuitive method to find link or node disjoint paths in a network is based on the shortest paths algorithm. The paths are obtained iteratively: once a shortest path between a pair of nodes is found, its links or nodes, respectively, are removed from the topology. When no additional path can be found, the algorithm stops. This simple approach does not always lead to the shortest disjoint paths (see Figure 2(b)) or it cannot even find disjoint paths (see Figure 2(a)) at all although disjoint paths exist (see Figure 2(c)).

2) Algorithms for Shortest Disjoint Paths Calculation: Bhandari's book [19] gives a good overview on different algorithms to find disjoint paths in networks. Several algorithms find pairs of disjoint shortest paths [20]. This is, however, not yet sufficient for the path layout of the SPM since its multipath should be as broad as possible. The k-disjoint shortest paths algorithm [21] yields k disjoint shortest paths if k such paths exist in the topology, otherwise the algorithm returns the maximum set of disjoint shortest paths. Node disjoint paths can be found with the same algorithm as for link disjoint paths if the underlying graph structure is transformed in such a way that the nodes are represented by additional unidirectional links. In this work, we try to find at most 5 link and node disjoint paths for the path layout of the SPMs.

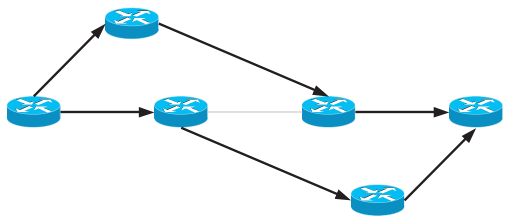
3) Adaptation to SRLGs: Shared risk link groups (SRLGs) are sets of links in a network that may fail simultaneously. Reasons may be, e.g., links on different wavelengths within a common fiber or links on different fibers within a common duct – they fail together in case of an electronic device failure or fiber cut. Another frequent reason for SRLGs are router



(a) First path prohibits second path.



(b) Suboptimal disjoint paths solution.



(c) Disjoint paths solution.

Fig. 2. The iterative application of the shortest path algorithms cannot find shortest disjoint paths in some cases.

failures. To work with SRLGs, the disjoint paths of SPMs should not contain links of the same SRLGs; otherwise, several paths of the SPM fail simultaneously and they do not protect each other anymore. Therefore, an adaptation of the paths layout to SRLGs needs to avoid links of common SRLGs on disjoint paths. This is a difficult NP-hard problem [22] which cannot be solved efficiently for general SRLGs. However, specific SRLGs can be respected efficiently, e.g. by node disjoint paths like in this work. The path layout for SPMs in case of SRLGs is not the focus of our work but rather the optimization of the path failure specific load balancing functions for SPMs in the next section.

B. Optimization of the Load Balancing Functions

The objective of this section is the optimization of the path failure specific load balancing functions for SPMs. First, we explain our notation of path concepts, then we introduce implications of failure scenarios, and finally, we propose two simple heuristics and the optimization for the load balancing functions to reduce the overall required network capacity.

1) Notation of Path Concepts: We introduce some basic notation from linear algebra that we use to model links, traffic aggregates, single paths, and multipaths.

a) Basic Notation from Linear Algebra: Let $\mathbb{X}$ be a set of elements, then $\mathbb{X}^n$ is the set of all n-dimensional vectors and $\mathbb{X}^{n\times m}$ the set of all $n\times m$-matrices with components taken from $\mathbb{X}$. Vectors $\mathbf{x}\in\mathbb{X}^n$ and matrices $\mathbf{X}\in\mathbb{X}^{n\times m}$ are written bold and their components are written as $\mathbf{x}=\begin{pmatrix} x_0 \\ \vdots \\ x_{n-1}\end{pmatrix}$ and $\mathbf{X}=\begin{pmatrix} x_{0,0} & \cdots & x_{0,m-1} \\ \vdots & & \vdots \\ x_{n-1,0} & \cdots & x_{n-1,m-1}\end{pmatrix}$. The scalar multiplication $c\cdot\mathbf{v}$ and the transpose operator $^\top$ are defined as usual. The scalar product of two n-dimensional vectors $\mathbf{u}$ and $\mathbf{v}$ is written with the help of matrix multiplication $\mathbf{u}^\top\mathbf{v}=\sum_{i=1}^{n}u_i v_i$. Binary operators $\circ\in\{+,-,\cdot\}$ are applied component-wise, i.e. $\mathbf{u}\circ\mathbf{v}=(u_0\circ v_0,\ldots,u_{n-1}\circ v_{n-1})^\top$. The same holds for relational operators $\circ\in\{<,\leq,=,\geq,>\}$, i.e. $\mathbf{u}\circ\mathbf{v}$ equals $\forall\ 0\leq i<n\colon u_i\circ v_i$. For simplicity reasons we define special vectors $\mathbf{0}=(0,\ldots,0)^\top$ and $\mathbf{1}=(1,\ldots,1)^\top$ with context specific dimensions.

b) Links and Nodes: A network $\mathcal{N}=(\mathcal{V},\mathcal{E})$ consists of $n=|\mathcal{V}|$ nodes and $m=|\mathcal{E}|$ unidirectional links. The links are represented as unit vectors $\mathbf{e_i}\in\{0,1\}^m$, i.e.

$$(e_i)_j=\begin{cases}0 & i\neq j\\ 1 & i=j\end{cases}\ \text{for } 0\leq i,j<m.$$

c) Traffic Aggregates: We denote traffic aggregates between routers $\mathbf{v_i}$ and $\mathbf{v_j}$ by $d=(i,j)$ and the set of all aggregates by $\mathcal{D}=\{(i,j):0\leq i,j<n \text{ and } i\neq j\}$.

d) Single Paths: A single path p_d associated with a demand $d\in\mathcal{D}$ between two distinct nodes is a set of contiguous links represented by a link vector $\mathbf{p_d}\in\{0,1\}^m$.

e) Multipaths: The basic structure of an SPM for a traffic aggregate d is a multipath $\mathbf{P_d}$ that consists of k_d paths $\mathbf{p_d^i}$ for $0\leq i<k_d$ that are link and possibly also node disjoint except for their source and destination nodes. It is represented by a vector of single paths $\mathbf{P_d}=(\mathbf{p_d^0},...,\mathbf{p_d^{k_d-1}})$. Thus, a multipath is described by a matrix $\mathbf{P_d}\in\{0,1\}^{k_d\times m}$.

2) Implications of Failure Scenarios: We first define the set of protected failure scenarios and then we discuss the implications of failure scenarios in general.

a) Set of Protected Failure Scenarios: A failure scenario s is given by a set of failed links and nodes. The set of protected failure scenarios $\mathcal{S}$ contains all outage cases including the normal working case for which the SPM should protect the traffic from being lost.

b) Failure Indication Function $\phi(\mathbf{p},s)$ for a Single Path: The failure indication function $\phi(\mathbf{p},s)$ yields one if a path p is affected by a failure scenario s; otherwise it yields zero.

c) Failure Symptom $\mathbf{f_d(s)}$ for a Multipath: The failure symptom of a multipath $\mathbf{P_d}$ is the vector $\mathbf{f_d(s)}=\left(\phi(\mathbf{p_d^0},\mathbf{s}),...,\phi(\mathbf{p_d^{k_d-1}},\mathbf{s})\right)^\top$ and indicates its failed single paths in case of failure scenario s. Thus, with a failure symptom of $\mathbf{f_d}=\mathbf{0}$, all paths are working while for $\mathbf{f_d}=\mathbf{1}$ connectivity cannot be maintained. In this work, we take the protection of all single link or node failures into account such that at most one single path of an SPM multipath fails. The set of all different failure symptoms for the SPM $\mathbf{P_d}$ is denoted by $\mathcal{F}_d=\{\mathbf{f_d(s)}:s\in\mathcal{S}\}$.

d) Traffic Reduction due to Failure Scenarios: Normally, all traffic aggregates $d\in\mathcal{D}$ are active. If routers fail, some demands disappear which leads to a traffic reduction that is expressed by the failure scenario specific set of aggregates $\mathcal{D}_s$.

- *No Traffic Reduction (NTR):* We assume hypothetically that failed routers lose only their transport capability for transit flows but they are still able to generate traffic. Therefore, we have $\mathcal{D}_s = \mathcal{D}$.

- *Source Traffic Reduction (STR):* If a certain router fails, all traffic aggregates with this source node disappear.
- *Full Traffic Reduction (FTR):* We assume that traffic aggregates with failed source *or* destination are stalled.

We use FTR for the computation of the results in this paper but we considered all options in [23].

3) The Load Balancing Function and Simple Heuristics: We introduce the load balancing function for the SPM and present two simple heuristics for their configuration.

a) Load Balancing Function $\mathbf{l}_\mathbf{d}^\mathbf{f}$*:* There is one SPM for each traffic aggregate $d \in \mathcal{D}$. This SPM has a load balancing function to distribute the traffic over its k_d different paths. If certain paths fail, which is indicated by the symptom $\mathbf{f_d}(\mathbf{s})$, the load balancing function shifts the traffic to the remaining working paths. Thus, the SPM needs a load balancing function $\mathbf{l}_\mathbf{d}^\mathbf{f}$ for each symptom $\mathbf{f} \in \mathcal{F}_d$ that results from any protected failure scenarios $s \in \mathcal{S}$. Since the load balancing function $\mathbf{l}_\mathbf{d}^\mathbf{f} \in (\mathbb{R}_0^+)^{k_d}$ describes a distribution, it must obey

$$\mathbf{1}^\top \mathbf{l}_\mathbf{d}^\mathbf{f} = 1. \qquad (1)$$

Furthermore, failed paths must not be used, i.e.

$$\mathbf{f}^\top \mathbf{l}_\mathbf{d}^\mathbf{f} = 0. \qquad (2)$$

Two very simple non-optimized options for load balancing are presented subsequently.

b) Equal Load Balancing: The traffic may be distributed equally over all working paths, i.e., $\mathbf{l}_\mathbf{d}^\mathbf{f} = \frac{1}{\mathbf{1}^\top(\mathbf{1}-\mathbf{f})} \cdot (\mathbf{1} - \mathbf{f})$.

c) Reciprocal Load Balancing: The load balancing factors may be indirectly proportional to the length of the partial paths $(\mathbf{1}^\top \mathbf{p}_\mathbf{d}^\mathbf{i})$. This can be computed by $(l_d^f)_i = \frac{1-f_i}{\mathbf{1}^\top \mathbf{p}_\mathbf{d}^\mathbf{i}} / \left(\sum_{0 \le j < k_d} \frac{1-f_j}{\mathbf{1}^\top \mathbf{p}_\mathbf{d}^\mathbf{j}}\right)$. These very simple heuristics require a lot of backup capacity [2]. Therefore, optimization of the load balancing function is required.

4) Optimization of the Load Balancing Function: The optimization configures the load balancing function in such a way that the required network bandwidth is minimal for a given traffic matrix. We first introduce the traffic matrix and the link bandwidths. Then, we formulate capacity constraints to guarantee sufficient resources in all protected failure scenarios. Finally, we summarize the linear program and formulate the objective function that allows the calculation of optimized load balancing functions.

a) Traffic Matrix and Link Bandwidths: The traffic rate associated with each traffic aggregate $d \in \mathcal{D}$ is given by $c(d)$ and corresponds to an entry in the traffic matrix. We describe the network capacity by a bandwidth vector $\mathbf{b} \in (\mathbb{R}_0^+)^m$, which carries a capacity value for each link. The overall capacity in the network can be calculated by $\mathbf{1}^\top \mathbf{b}$. Similarly, the vector indicating the traffic rates on all links, which are induced by a specific SPM $\mathbf{P_d}$ and a specific failure symptom $f \in \mathcal{F}_d$, is calculated by $\mathbf{P_d} \cdot \mathbf{l}_\mathbf{d}^\mathbf{f} \cdot c(d)$.

b) Capacity Constraints with Bandwidth Reuse: We now formulate capacity constraints that describe the required link bandwidths which are necessary to support the traffic in all protected scenarios $s \in \mathcal{S}$. In packet switched networks, resources are not physically bound to traffic aggregates. If traffic is rerouted due to a local outage, the released resources can be automatically reused for the transport of other traffic. Under this assumption, the capacity constraints are

$$\forall s \in \mathcal{S} : \sum_{d \in \mathcal{D}_s} \mathbf{P_d} \cdot \mathbf{l}_\mathbf{d}^{\mathbf{f_d(s)}} \cdot c(d) \le \mathbf{b}. \qquad (3)$$

In [2], [23], we have also proposed and investigated constraints that apply when capacity cannot be reused.

c) Linear Program for SPM Optimization: The objective of the optimization is the minimization of the required network bandwidth. Thus, the objective function is

$$\mathbf{1}^\top \mathbf{b} \to \min. \qquad (4)$$

The free variables, that must be set in the optimization process, are the load balancing functions and the link bandwidths.

$$\forall d \in \mathcal{D}\ \forall \mathbf{f} \in \mathcal{F}_d : \mathbf{l}_\mathbf{d}^\mathbf{f} \in (\mathbb{R}_0^+)^{k_d} \qquad (5)$$

$$\mathbf{b} \in (\mathbb{R}_0^+)^m. \qquad (6)$$

The following constraints must be respected in the optimization process to obtain valid load balancing functions and feasible link bandwidths.

(C0) Equation (1) assures that the load balancing function is a distribution.

(C1) Equation (2) assures that failed paths will not be used.

(C2) Equation (3) assures that the bandwidth suffices to carry the traffic in all protected failure scenarios.

C. Analysis of the Linear Program Size

The runtime of the above LP depends on the number of free variables, the additional constraints, and the structure of the program. We briefly estimate them depending on the network size.

1) Number of Free Variables: The link bandwidth $\mathbf{b}$ comprises exactly $n_1^{free} = m$ free variables.

The consideration of the load balancing function $\mathbf{l}_d^{f_d(s)}$ is more complex. One SPM exists for each traffic aggregate $d \in \mathcal{D}$ and for each SPM a load balancing function $\mathbf{l}_\mathbf{d}^\mathbf{f}$ is needed for every SPM failure symptom $f \in \mathcal{F}_d$. A load balancing vector has an entry for each of the k_d link and node disjoint paths of the SPM. There is one load balancing vector for each SPM failure symptom. We take all single link and node failures into account in addition to the working scenario, so we have exactly $|\mathcal{F}_d| = k_d + 1$ different failure symptoms. We use a full traffic matrix in our study, thus, the number of traffic aggregates is $|\mathcal{D}| = n \cdot (n-1)$. We denote the average number of outgoing links per node by the average node degree deg_{avg} which can be calculated by $deg_{avg} = \frac{m}{n}$. The average number of disjoint paths for all SPMs is given by $k^* = \frac{1}{|\mathcal{D}|} \cdot \sum_{d \in \mathcal{D}} k_d$ and it is smaller than the average node degree $k^* \le deg_{avg}$. Taking this into account, the overall number of free variables is $n_2^{free} = \sum_{d \in \mathcal{D}} k_d \cdot (k_d + 1) \approx n \cdot (n-1) \cdot k^* \cdot (k^* + 1) \leq m^2$. Thus, the number of free variables $n_1^{free} + n_2^{free} \approx m + m^2$ scales quadratically with the number of links in the networks.

2) Number of Constraints: We calculate the number of constraints resulting from (C0), (C1), and (C2) in Section III-B.4.c. Both (C0) and (C1) require for each path failure specific load balancing function one constraint such that we get $n_{C0} = n_{C1} = \sum_{d \in \mathcal{D}} (k_d + 1) \approx n \cdot m$ different equations. Constraint type (C2) requires an equation for each protected failure scenario (working scenario and all single link and node failures) and for each link. Thus, the number of constraints is exactly $n_{C2} = (1 + m + n) \cdot m$. After all, we get an overall number of constraints of at most $n_{cnst}^{all} = m^2 + 3 \cdot m \cdot n + m$. Thus, the number of constraints also scales about quadratically with the number of links in the network.

IV. Results

In this section, we show first the efficiency of the SPM as protection switching mechanism. Then, we illustrate the computation time and the memory requirements of the above described optimization algorithm for two different LP solving approaches and illustrate the dependency of the computation time on the network structure.

A. Efficiency of the SPM as a Protection Switching Algorithm

We show by means of a multitude of sample networks that the SPM is a very efficient protection switching mechanism. First, we calculate the required capacity for networks dimensioned for shortest path routing without any protection since this combination requires the least capacity in a network. Then, we calculate the required capacity for the SPM with protection of all single link and node failures and consider the additionally needed network capacity as backup capacity. We use the backup capacity as the performance criterion in this study. First, we illustrate the required backup capacity for random networks depending on different topological characteristics. Then, we have a look at a specific research network and compare the required backup for the SPM and for p-cycles.

1) Dependency of the Backup Capacity on Topological Network Characteristics: In this section, we investigate the required backup capacity depending on topological network characteristics. The degree of a network node is the number of its outgoing links. We construct sample networks for which we control the number of nodes $n \in \{10, 15, 20, 25, 30\}$, the average node degree $deg_{avg} \in \{3, 4, 5, 6\}$, and the maximum deviation of the individual node degree from the average node degree $deg_{dev}^{max} \in \{1, 2, 3\}$. We use the algorithm of [2] for the construction of these networks since we cannot control these parameters rigidly with the commonly used topology generators [24]–[28]. We sampled 5 networks for each of the 60 different network characteristics and tested altogether 300 different networks. Figure 3 shows their required network capacity under the assumption of a homogenous traffic matrix. The network characteristics determine the shape of the points, the corresponding x-coordinate values indicate the average number of disjoint paths k^* for the SPMs in the network, and the y-coordinate values give us the required backup capacity. Each point corresponds to the average of 5 networks with the same characteristics.

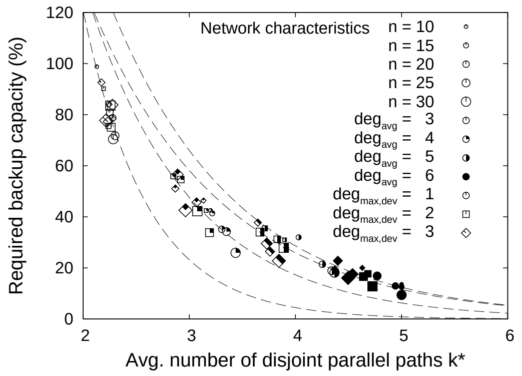


Fig. 3. Average backup capacity requirements for random networks depending on their average number of average parallel paths.

There is an obvious trend in the figure: the required backup capacity decreases significantly with an increasing number of parallel paths k^* for the SPMs. Networks with the same average node degree deg_{avg} are obviously clustered, which results from the fact that the average node degree deg_{avg} and k^* are strongly correlated. The dashed lines represent an exponential extrapolation based on a least square approximation of the points in each cluster. Such a cluster contains networks with different deviations deg_{dev}^{max} of individual node degrees from the average node degree deg_{avg}. Those networks with a small deviation deg_{dev}^{max} have a larger k^* than those with a large deg_{dev}^{max} and need, therefore, less backup capacity. Large networks require slightly less capacity than small networks, however, this trend is not so obvious. The SPM is quite efficient since 20% backup capacity suffice to protect the traffic against all single link and node failures, provided that the networks allow for enough disjoint paths.

Shortest paths rerouting is another option to find backup paths which can be used for restoration. In [1], [2] we have shown that this requires significantly more additional capacity than the SPM, and that the superiority of the SPM over shortest path routing increases for networks with more disjoint paths. Obviously, the SPM can take advantage of a highly meshed network topology while shortest path rerouting cannot profit from it regarding its required backup capacity.

2) Comparison of the Efficiency of SPM and p-Cycles: In [14] the p-cycle concept has been investigated. An optimal p-cycle layout has been found to protect the network with the least capacity possible using a maximum cycle length as side constraint. The experiments were conducted with the COST-239 network, which has been a pan-European research network that is often used in literature together with its original, partly asymmetric traffic matrix [29], [30]. The most effective solution required 44% backup capacity for p-cycles while the SPM requires only 23.4%. After all, the SPM is a very efficient protection switching algorithm.

B. Experimental Runtime Analysis of the Optimization Algorithm

We study the runtime requirements of the optimization algorithm both with respect to memory consumption and computation time since this has a tremendous impact on the feasibility of the above proposed optimization approach. First, we give a short introduction to linear programs (LP) and solvers as well as to our implementation. Then, we analyze the memory consumption and the computation time of different LP solvers and, finally, we illustrate the computation time for the above mentioned results.

1) Linear Programs and Testbed: The solutions of LPs may consist of rational numbers, they may be restricted to integer solutions, then the problems are called Integer (Linear) Programs (IP, ILP), or they may be partly restricted to integer solutions, then the problems are called Mixed Integer (Linear) Programs (MIP, MILP)) [31]. ILPs or MILPs are NP-complete problems. Fortunately, our LP formulation has a rational solution. Therefore, it can be solved by the Simplex algorithm or by Interior point methods (IPMs). The Simplex algorithm is quite fast in general, but it may have an exponential runtime in the worst case. In contrast, IPMs run in polynomial time [32] but they are more complex. Our implementation of the above LP uses the free software *GNU Linear Programming Kit* version 4.8 [33] as an LP solver, which offers a Simplex- and an IPM-based solver. Due to license issues, we avoided commercial standard software. We used an Intel Pentium 4 with a CPU of 3.20 GHz and 2 GB RAM. The operating system is SuSE 9.1. We used the ICC 8.1 compiler [34] with options "-O3 -xP". Other compilers like GCC 3.3.3 [35] led to the same memory consumptions but to longer computation times.

2) Memory Consumption and Computation Time of Different LP Solvers: First, we consider the efficiency of the above mentioned solver methods for LPs of different size. We take sample networks with 10, 15, 20, 25, and 30 nodes with an average node degree of $deg_{avg} = 5$ and a maximum deviation of $deg_{dev}^{max} = 3$. We calculate the solution of the LP and monitored the memory consumption and the computation time. Figure 4 shows the results. The computation time for the optimization program based on the Simplex method is about 100 to 150 times faster than the one based on IPM. This holds both for small problems in networks with only 10 nodes that execute within 1 second and for large problems in medium-size networks with 30 nodes that execute within 16 minutes for the Simplex method. The memory consumption is 4.7 MB for small problems and 139 MB for large problems if the program is based on the Simplex method. For IPM, the memory consumption is 4.5 to 13.4 times larger and the memory savings of the Simplex method increase with problem complexity. After all, the Simplex method is more appropriate than IPM to solve this specific minimization problem. Therefore, we use it as the default solver in our studies.

3) Computation Time for Random Networks Using the Simplex Method: Above, we have studied the runtime requirements for four different networks to identify the appropriate

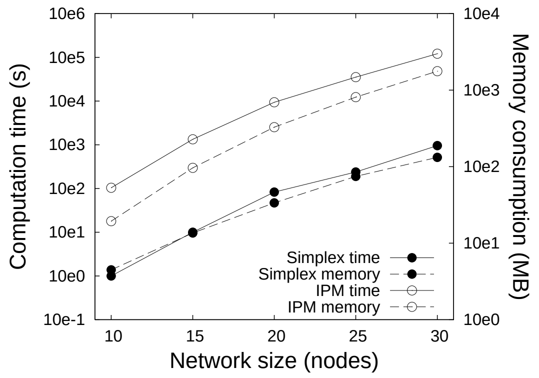


Fig. 4. Comparison of memory consumption and computation time for the Simplex and an interior point method for networks of different size.

LP solver for the implementation of the minimization problem. Now, we investigate the computation time depending on the network characteristic. In particular, we analyze the runtime for the networks considered in Section IV-A.1.

Figure 5 shows the average computation time depending on some network characteristics. Each point in the graph denotes the average computation time of 15 networks with a common number of nodes and a common number of links, i.e., these networks have also a common node degree. The computation time increases both with the number of links and the number of nodes in a network. For networks with a certain number of links, the average computation time varies not more than by a factor of 10. In contrast, for networks with a certain number of nodes, the average computation time varies over several orders of magnitude. Thus, the number of links dominates the computation time of the optimization program.

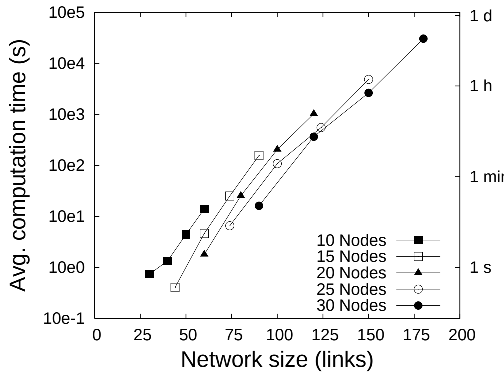


Fig. 5. Computation time for the optimization of different random networks.

V. Conclusion

In this paper, we have reviewed several protection switching mechanisms and, in particular, the self-protecting multipath (SPM). Its structure is composed of disjoint paths that can be calculated by a shortest disjoint paths algorithm. The traffic is distributed over these paths according to a load balancing function that can be optimized in such a way that the sum of all link capacities which are required to carry the traffic in all protected failure scenarios is minimized. We presented the optimization algorithm for the load balancing function which is based on a linear program (LP).

We performed a numerical study based on random and existing networks and took into account the protection of all single link and node failures. We showed that the SPM is a very efficient protection switching mechanism if sufficiently many disjoint paths can be found in a network: additional backup capacity in the order of 20% may be enough to protect the traffic against all single link and node failures. The efficiency is clearly better than shortest path rerouting and the consideration of the COST-239 network showed that the SPM also outperforms the p-cycle approach. We first analyzed the complexity of the LP theoretically and then illustrated its computation time and memory consumption experimentally. The program complexity is dominated by the number of links in the network. The LP solution based on the Simplex algorithm requires less memory and runs considerably faster than the LP solution based on interior point methods (IPM) although the Simplex algorithm has an exponential-time runtime complexity in the worst case whereas IPM runs within polynomial-time.

After all, the SPM is a capacity-efficient and simple protection switching mechanism and, therefore, its application in practice is of interest. However right now, the SPM is only applicable in small and medium size networks due to its computation time and memory demand. Hence, suitable heuristics are required to minimize the objective function of the LP and to optimize thereby the load balancing functions of the SPM in large networks. Furthermore, we are currently working on an optimized configuration of the SPM for networks with given link capacities and traffic matrix to minimize the maximum link utilization. For large networks, this problem also requires heuristic approaches.

REFERENCES

[1] M. Menth, A. Reifert, and J. Milbrandt, "Self-Protecting Multipaths - A Simple and Resource-Efficient Protection Switching Mechanism for MPLS Networks," in 3rd *IFIP-TC6 Networking Conference (Networking)*, Athens, Greece, May 2004, pp. 526 – 537.

[2] M. Menth, "Efficient Admission Control and Routing in Resilient Communication Networks," PhD thesis, University of Würzburg, Faculty of Computer Science, Am Hubland, July 2004.

[3] A. Nucci, B. Schroeder, S. Bhattacharyya, N. Taft, and C. Diot, "IGP Link Weight Assignment for Transient Link Failures," in 18th *International Teletraffic Congress (ITC)*, Berlin, Sept. 2003.

[4] B. Fortz and M. Thorup, "Robust Optimization of OSPF/IS-IS Weights," in *International Network Optimization Conference (INOC)*, Paris, France, Oct. 2003, pp. 225–230.

[5] A. Autenrieth and A. Kirstädter, "Engineering End-to-End IP Resilience Using Resilience-Differentiated QoS," *IEEE Communications Magazine*, vol. 40, no. 1, pp. 50–57, Jan. 2002.

[6] P. Pan, G. Swallow, and A. Atlas, "RFC4090: Fast Reroute Extensions to RSVP-TE for LSP Tunnels," May 2005.

[7] H. Saito and M. Yoshida, "An Optimal Recovery LSP Assignment Scheme for MPLS Fast Reroute," in *International Telecommunication Network Strategy and Planning Symposium (Networks)*, June 2002, pp. 229–234.

[8] A. Kvalbein, A. F. Hansen, T. Cicic, S. Gjessing, and O. Lysne, "Fast Recovery from Link Failures Using Resilient Routing Layers," in *IEEE Symposium on Computers and Communications (ISCC)*, Cartagena, Spain, June 2005.

[9] M. Menth and R. Martin, "Network Resilience through Multi-Topology Routing," in *The* 5th *International Workshop on Design of Reliable Communication Networks*, Island of Ischia (Naples), Italy, Oct. 2005, pp. 271 – 277.

[10] M. Shand and S. Bryant, "IP Fast Reroute Framework," http://www.ietf.org/internet-drafts/draft-ietf-rtgwg-ipfrr-framework-04.txt, Oct. 2005.

[11] A. Atlas and A. Zinin, "Basic Specification for IP Fast-Reroute: Loop-Free Alternates," http://www.ietf.org/internet-drafts/draft-ietf-rtgwg-ipfrr-spec-base-04.txt, July 2005.

[12] D. Stamatelakis and W. D. Grover, "IP Layer Restoration and Network Planning Based on Virtual Protection Cycles," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, Oct. 2000.

[13] D. A. Schupke, "Automatic Protection Switching for p-Cycles in WDM Networks," *Optical Switching and Networking: A Computer Networks Journal*, vol. 2, no. 1, pp. 35 – 48, May 2005.

[14] C. G. Gruber and D. A. Schupke, "Capacity-Efficient Planning of Resilient Networks with p-Cycles," in *International Telecommunication Network Strategy and Planning Symposium (Networks)*, June 2002, pp. 389–395.

[15] B. Fortz, J. Rexford, and M. Thorup, "Traffic Engineering with Traditional IP Routing Protocols," *IEEE Communications Magazine*, 2002.

[16] S. Köhler and A. Binzenhöfer, "MPLS Traffic Engineering in OSPF Networks - A Combined Approach," in 18th *International Teletraffic Congress (ITC)*, Berlin, Germany, Sept. 2003.

[17] M. Pióro and D. Medhi, *Routing, Flow and Capacity Design in Communication and Computer Networks*. Morgan and Kaufman, June 2004.

[18] K. Murakami and H. S. Kim, "Optimal Capacity and Flow Assignment for Self–Healing ATM Networks Based on Line and End-to-End Restoration," *IEEE/ACM Transactions on Networking*, vol. 6, no. 2, pp. 207–221, Apr. 1998.

[19] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing*. Norwell, MA, USA: Kluwer Academic Publishers, 1999.

[20] J. W. Suurballe, "Disjoint Paths in a Network," *Networks Magazine*, vol. 4, pp. 125–145, 1974.

[21] J. Edmonds and R. M. Karp, "Theoretical Improvements in the Algorithmic Efficiency for Network Flow Problems," *Journal of the ACM*, vol. 19, no. 2, pp. 248–264, Apr. 1972.

[22] J. Q. Hu, "Diverse Routing in Optical Mesh Networks," *IEEE/ACM Transactions on Networking*, vol. 51, no. 3, pp. 489 – 494, 2003.

[23] M. Menth, J. Milbrandt, and A. Reifert, "Sensitivity of Backup Capacity Requirements to Traffic Distribution and Resilience Constraints," in 1st *Conference on Next Generation Internet Networks Traffic Engineering (NGI)*, Rome, Italy, Apr. 2005.

[24] B. M. Waxman, "Routing of Multipoint Connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.

[25] E. W. Zegura, K. L. Calvert, and M. J. Donahoo, "A Quantitative Comparison of Graph-Based Models for Internet Topology," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 770–783, 1997.

[26] C. Jin, Q. Chen, and S. Jamin, "Inet: Internet Topology Generator," Department of EECS, University of Michigan, USA, Tech. Rep. CSE-TR-433-00, 2000.

[27] A. Medina, I. Matta, and J. Byers, "BRITE: An Approach to Universal Topology Generation," in *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Cinncinnati, Ohio, USA, Aug. 2001.

[28] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger, "Network Topology Generators: Degree-Based vs. Structural," in *ACM SIGCOMM*, Aug. 2002.

[29] P. Batchelor et al., "Ultra High Capacity Optical Transmission Networks. Final Report of Action COST 239," Jan. 1999.

[30] C. Mauz, "Mapping of Arbitrary Traffic Demand and Network Topology on a Mesh of Rings Network," in *IFIP Working Conference on Optical Network Design and Modelling*, Feb. 2001.

[31] Optimization Technology Center of Northwestern University and Argonne National Laboratory, "Linear Programming Frequently Asked Questions," http://www-unix.mcs.anl.gov/otc/Guide/faq/linear-programming-faq.html, Sept. 2005.

[32] N. Karmakar, "A New Polynomial-Time Algorithm for Linear Programming," *Combinatorica*, vol. 4, pp. 373 – 396, 1984.

[33] The Free Software Foundation, "The GNU Linear Programming Kit (GLPK) Version 4.8," http://www.gnu.org/software/glpk/glpk.html, Boston, MA, USA, 2005.

[34] Intel Corporation, "The Intel C Compiler," http://www.intel.com/cd/software/products/asmo-na/eng/compilers/, Santa Clara, CA, USA, 2005.

[35] The Free Software Foundation, "The GNU Compiler Collection (GCC)," http://www.gnu.org/software/gcc/gcc.html, Boston, MA, USA, 2005.