Robust IP Link Costs for Multilayer Resilience

Michael Menth, Matthias Hartmann, and Rüdiger Martin

University of Würzburg, Institute of Computer Science, Germany
{menth,hartmann,martin}@informatik.uni-wuerzburg.de

Abstract. In this work we optimize administrative link costs of IP networks in such a way that the maximum utilization of all links is as low as possible for a set of considered failure scenarios (e.g., all single link failures). To that aim, we present the new "hill hopping" heuristic with three different variants and compare their computation times and the quality of their results. We adapt the objective function of the heuristic to make the link cost settings robust to single link failures, single node failures, and single link or node failures, and compare the results. In particular, we optimize the routing for multilayer networks where unused backup capacity of the link layer can be reused to redirect traffic on the network layer in case of an IP node failure.

1 Introduction

IP routing is very robust against network failures as it always finds possible paths between two endpoints as long as they are still physically connected. When a failure occurs, traffic is rerouted which may lead to congestion on the backup paths. In fact, this is the most frequent cause for overload in IP backbones [1] and may violate the quality of service (QoS) in terms of packet loss and delay.

In IP networks, traffic is forwarded along least-cost paths whose costs are based on the sum of the administrative costs of their links. The modification of the administrative link costs changes the routing and is thereby a means for traffic engineering. The link costs are usually set to one, which is the hop count metric, proportionally to the link delay, or reciprocally to the link bandwidth. However, for a network with a given topology, link bandwidths, and traffic matrix, the maximum link utilization can be minimized by choosing appropriate link costs, but this problem is NP-hard [2]. Therefore, heuristic methods are applied to solve it [3].

In the presence of failures, the overload due to backup traffic may be reduced by influencing the routes by a modification of the link costs. Calculating new costs and uploading them on the routers takes some time and this is cumbersome because most outages last less than 10 minutes [4]. In addition, when the failed link resumes operation, the new link costs may be suboptimal. A simpler solution is setting the link costs a priori in such a way that they lead to a low utilization of all links both under failure-free conditions and after rerouting for a set of protected failures $\mathcal{S}$. So far, only a few papers [5–8] addressed this kind of optimization and and they considered only the protection of single link failures.

This work was funded by Siemens AG, Munich, and by the Deutsche Forschungsgemeinschaft (DFG) under grant TR257/23-1. The authors alone are responsible for the content of the paper.

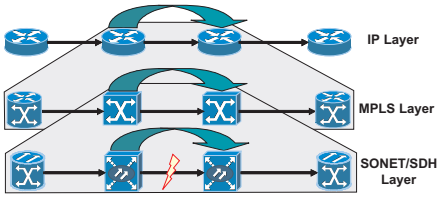


Fig. 1. Connections of lower layers provide links for upper layers.

In multilayer networks, connections of lower layers provide links for upper layers. Figure 1 illustrates that a logical IP network link may be implemented by a label switched path (LSP) of an underlying MPLS layer. This LSP contains further intermediate label switching routers (LSRs) not visible on the IP layer. Likewise, links between these LSRs may be implemented by virtual or physical connections of an underlying SONET/SDH or optical layer. Multilayer networks provide rerouting or protection switching capabilities and backup capacity on different layers [9]. This seems to be a waste of resources, but protection on lower layers reacts faster than rerouting on the IP layer. To save bandwidth, it is desirable to share the backup capacity between layers, which is possible between the IP layer and the packet-switched MPLS layer.

The contribution of this paper is manifold. It suggests the new "hill hopping" heuristic for the optimization of resilient IP routing with three different intuitive variants. It presents a new methodology for an empirical comparison of the computation times of the algorithms and the quality of their results. And it goes beyond the protection of single link failures as node failures are also considered and, in particular, backup capacity sharing between layers for multilayer networks.

Section 2 gives the problem formulation for resilient IP routing and summarizes related work. Section 3 proposes several new heuristics and compares their computation times and the quality of their results. Section 4 adapts the objective function of the heuristics to different protection variants including multilayer protection and illustrates their impact on the bandwidth efficiency. Finally, we summarize this work and draw our conclusions in Section 5.

2 Optimization of IP Routing with and without Resilience Requirements

In this section, we review fundamentals of IP routing and summarize related work on routing optimization with and without resilience requirements.

2.1 Fundamentals of IP Routing

In IP networks, routers have routing tables that contain for many IP-address-prefixes one or several next hops. A router forwards an incoming packet by finding the longest prefix in the routing table that matches its destination address and by sending it to one of the corresponding next hops. Thus, IP implements destination-based routing. Single path routing forwards the traffic only to the interface or next hop with, e.g., the lowest ID while multipath routing splits the traffic equally among all possible next hops (cf. 7.2.7 of [10]). The routing tables are usually constructed in a distributed manner by routing protocols like OSPF or IS-IS that exchange information about the current topological structure of the network. A router calculates the next hops to all other routers in the network by using the shortest (or least-cost) paths principle to avoid forwarding loops. In particular, sink trees are computed to every destination in the network. In addition a router knows which node in the network serves as egress router to prefixes outside the network. This combined information is constructed into the routing table. IP routing

is very robust against network failures because in case of a failure, the new topological information is exchanged by the routing protocol and routers update their routing tables. This rerouting may take seconds, but currently new mechanisms for IP fast rerouting are investigated [11]. Single-path routing is default, but we apply the equal-cost multi-path (ECMP) option, which allows multi-path routing over all least-cost paths towards the same destination. It makes the routing independent of device numbers and enables fast and local traffic redirection if a next hop fails and several next hops exist [12].

2.2 Problem Formulation

We model a network by a graph $G = (\mathcal{V}, \mathcal{E})$ consisting of its set of nodes $\mathcal{V}$ and its set of directed links $\mathcal{E}$. Calligraphic letters $\mathcal{X}$ denote sets and the operator $|\mathcal{X}|$ indicates the cardinality of a set. Each link $l \in \mathcal{E}$ has a capacity $\mathbf{c}(l)$ and is associated with cost $\mathbf{k}(l)$. The capacities and the costs of all links are represented in a compact way by the vectors $\mathbf{c}$ and $\mathbf{k}$. Note that vectors and matrices are printed boldface and the indexed components of a vector $\mathbf{v}$ are denoted by $\mathbf{v}(i)$. We work with integer link cost between $k_{min} = 1$ and k_{max}, thus, they are taken from a vector space with $(k_{max})^{|\mathcal{E}|}$ elements.

A network is resilient to a certain failure scenario s if the rerouted traffic does not lead to congestion. Therefore, resilience always relates to a set of protected failure scenarios $\mathcal{S}$. Each $s \in \mathcal{S}$ describes a set of non-working network elements. For the sake of simple notation, the working scenario $\emptyset$ is part of $\mathcal{S}$. The function $u(l, v, w)$ indicates the percentage of the aggregate from node v to w that is carried over link l. This description models both single and multipath routing. We extend this routing function to $u_s^{\mathbf{k}}(l, v, w)$ to account for a specific set of link costs $\mathbf{k}$ and a certain failure scenario $s \in \mathcal{S}$. The traffic matrix $\mathbf{D}$ contains the demand rate $\mathbf{D}(v, w)$ between any two nodes $v, w \in \mathcal{V}$. The utilization $\rho(\mathbf{k}, l, s)$ of a link l in a failure scenario s, the maximum utilization $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l)$ of link l in all failure scenarios $s \in \mathcal{S}$, and the maximum utilization $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k})$ of all links $l \in \mathcal{E}$ in all failure scenarios $s \in \mathcal{S}$ is calculated for any link cost vector $\mathbf{k}$ by

$$\rho(\mathbf{k}, l, s) = \left(\sum_{v,w \in \mathcal{V}} u_s^{\mathbf{k}}(l, v, w) \cdot \mathbf{D}(v, w) \right) / c(l) \tag{1}$$

$$\rho_{\mathcal{S}}^{max}(\mathbf{k}, l) = \max_{s \in \mathcal{S}} \left(\rho(\mathbf{k}, l, s) \right) \tag{2}$$

$$\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k}) = \max_{l \in \mathcal{E}} \left(\rho_{\mathcal{S}}^{max}(\mathbf{k}, l) \right) \tag{3}$$

Note that the calculation of Equations (2) and (3) is quite costly since destination trees need to be calculated by Dijkstra's algorithm for each protected network failure $s \in \mathcal{S}$. When our algorithms calculate $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k}')$ to test for $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k}') < \rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k})$, the calculation of $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k}')$ stops as soon as $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k})$ has been exceeded to save computation time. In addition, the failure scenarios can be sorted in such a way that this condition occurs early.

The objective of IP routing optimization is to find a link cost vector $\mathbf{k}$ such that the maximum link utilization $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k})$ is minimal. If resilience is not required, the set of protected failure scenarios contains only the working scenario $\mathcal{S} = \{\emptyset\}$, otherwise it contains, e.g., all single (bidirectional) link failures.

2.3 Related Work

We briefly review existing work regarding the optimization of IP routing with and without resilience requirements.

Optimization of IP Routing without Resilience Requirements The problem of IP routing optimization without resilience requirements is NP-hard [2]. Some papers try to solve the problem by integer linear programs and branch and bound methods. Since the search space is rather large, others prefer fast heuristics and use local search techniques [3], genetic algorithms, simulated annealing, or other heuristics. The papers also differ slightly in their objective functions. In case of traffic hot spots or link failures, link costs may be changed, but this possibly causes service interruptions such that the number of changed link costs should be kept small [13].

Optimization of IP Routing with Resilience Requirements Optimization of IP routing becomes even more difficult if different failure scenarios must be taken into account for minimization of the objective function in Equation (3). It has been proposed independently by [5–7] for single link failures and almost at the same time. The presented algorithms use a local search technique combined with a tabu list or a hash function to mark already visited solutions. To escape from local minima, [5] sets some link weights to random values. To speed up the algorithm, [6] investigates only a random fraction of possible neighboring configurations while [7] applies an additional heuristic to generate a fraction of good neighboring configurations. Finally, [8] accelerates the evaluation of the objective function by considering only a set of critical links instead of the entire set of protected failure scenarios $\mathcal{S}$.

3 New Heuristics for Resilient IP Routing

In this section, we propose new heuristics to find good link costs $\mathbf{k}$ for resilient IP routing and compare their computation times and the quality of their results.

3.1 Description of the Algorithms

We apply the well-known hill climbing heuristic and propose the new hill hopping heuristic for resilient IP optimization. In addition, we propose three different methods for the generation of random neighbors $\mathbf{k_{new}}$ from a large neighborhood of the current link costs $\mathbf{k}$. These methods are required by hill hopping and can be reused by other heuristic control algorithms.

The Hill Climbing Algorithm The hill climbing algorithm starts with an initial current vector $\mathbf{k}$ of current link costs. It first evaluates the maximum link utilization $\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k_{new}})$ of all link cost vectors $\mathbf{k_{new}}$ in the close neighborhood of the current vector $\mathbf{k}$ which consists of all vectors that differ from $\mathbf{k}$ by at most 2 in a single link. It chooses the $\mathbf{k_{new}}$ with the best improvement $\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k}) - \rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k_{new}})$ as successor vector of $\mathbf{k}$. If no such $\mathbf{k_{new}}$ can be found, the algorithm terminates; otherwise, the procedure restarts with the new current vector $\mathbf{k}$.

The Hill Hopping Algorithm The quality of the results of the hill climbing algorithm suffers from the fact that it terminates when the first local minimum is found. We avoid this drawback by Algorithm 1. Here, the current cost vector $\mathbf{k}$ is substituted by $\mathbf{k_{new}}$ if its maximum utilization $\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k_{new}})$ is smaller than the one of the currently best link costs $\mathbf{k_{best}}$ multiplied by a factor $T \geq 1$. Thus, the maximum utilization of the current link costs can be slightly larger than $\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k_{best}})$. The method terminates if n^{unsuc}_{moves} new vectors $\mathbf{k_{new}}$ have been explored without finding a better one than $\mathbf{k_{best}}$.

In analogy to the hill climbing algorithm we call this method hill hopping. The current vector $\mathbf{k}$ has a high quality. We view this quality as a hill in the multi-dimensional

state space. A randomly generated successor $\mathbf{k_{new}}$ can be fairly distant from $\mathbf{k}$ and if it is accepted as new current vector $\mathbf{k}$, it also represents a quality hill. Thus, this method performs hill hopping. The design of this algorithm was inspired by the threshold accepting algorithm [14] which is a simplification of the simulated annealing heuristic.

Input: start vector $\mathbf{k_{start}}$, maximum number of unsuccessful moves n_{moves}^{unsuc}, threshold T for accepting new candidates
  $\mathbf{k} \leftarrow \mathbf{k_{start}}$, $\mathbf{k_{best}} \leftarrow \mathbf{k_{start}}$, $n \leftarrow 0$,
  while $n < n_{moves}^{unsuc}$ **do**
    $\mathbf{k_{new}} \leftarrow$GENERATERANDOMNEIGHBOR($\mathbf{k}$)
    $n \leftarrow n + 1$
    if $(\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k_{new}}) \leq T \cdot \rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k_{best}}))$ **then**
      $\mathbf{k} \leftarrow \mathbf{k_{new}}$
      if $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k}) < \rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k_{best}})$ **then**
        $\mathbf{k_{best}} \leftarrow \mathbf{k}$, $n \leftarrow 0$
      end if
    end if
  end while
Output: link costs $\mathbf{k_{best}}$

Algorithm 1: HILLHOPPING searches for link costs $\mathbf{k_{best}}$ that lead to low maximum link utilization $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k_{best}})$ in all protected failure scenarios $\mathcal{S}$.

Neighborhood Generation for the Hill Hopping Algorithm The hill hopping algorithm uses the method GENERATERANDOMNEIGHBOR for the generation of a new vector $\mathbf{k_{new}}$ in the wide neighborhood of $\mathbf{k}$. We propose three different implementations of that method.

Random Neighborhood Generation $RNG(h, d)$ The random neighborhood generation (RNG) randomly chooses h^* links according to a uniform distribution between 1 and h. It changes their costs by adding or subtracting an integral value between 1 and d, but the minimum cost value $k_{min} = 1$ and the maximum cost value k_{max} must be respected as side conditions.

Link Ranking Methods $r_{rel}^{\mathbf{k}}(l)$ and $r_{abs}^{\mathbf{k}}(l)$ The following neighborhood generation methods take advantage of the link-specific maximum utilization $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l)$ in Equation (2). The relative rank $r_{rel}^{\mathbf{k}}(l)$ of a link l is the number of links $l' \in \mathcal{E}$ that have a smaller utilization value $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l')$ than l. Note that several links possibly have the same relative rank. The absolute rank of a link $r_{abs}^{\mathbf{k}}(l)$ is its relative rank $r_{rel}^{\mathbf{k}}(l)$ plus the number of links l' with the same maximum link utilization $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l')$ but with a lower link ID than l. Both rankings yield numbers between 0 and $|\mathcal{E}|-1$. In contrast to the relative rank, the absolute rank is a 1:1 mapping.

Greedy Neighborhood Generation $GNG(h, d)$ The greedy neighborhood generation chooses a number h^* between 1 and h. It then chooses h^* links based on a special heuristic, and increases or decreases their costs if they carry high or little load, respectively. The heuristic to select the h^* links works as follows. The absolute rank $r_{abs}^{\mathbf{k}}(l)$ of a link l is associated with one of the $|\mathcal{E}|$ equidistant subintervals of $(0; 1)$ in Figure 2. A link is randomly chosen based on the probability density function $f(x) = (m + 1) \cdot (2 \cdot x - 1)^m$ in Figure 2. If the absolute rank $r_{abs}^{\mathbf{k}}(l)$ of that link l is smaller than $\frac{|\mathcal{E}-1|}{2}$, it has a relatively low maximum utilization value $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l)$. Therefore, its

cost is decreased by an integral random variable which is uniformly distributed between 1 and d. Otherwise it is increased by that value. This is repeated until the cost of h^* different links are changed. The GNG rarely changes the cost of links l with a medium maximum link utilization $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l)$ and it only increases (decreases) the costs of links with low (high) $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l)$. This is similar to the heuristic used in [7].

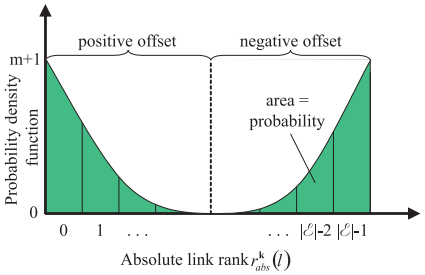


Fig. 2. The greedy neighborhood generation (GNG) chooses h^* links randomly according to the displayed probability density function using the absolute link rank $r_{abs}^{\mathbf{k}}$ and then modifies their costs by a negative or positive offset between 1 and d.

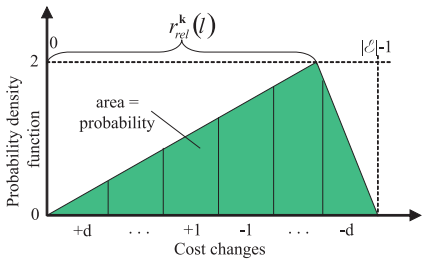


Fig. 3. The intelligent neighborhood generation (ING) chooses h^* links arbitrarily and then modifies their costs by an offset according to the displayed probability density function that depends on the relative rank $r_{rel}^{\mathbf{k}}(l)$ of the considered link.

Intelligent Neighborhood Generation $ING(h, d)$ Like the RNG, the intelligent neighborhood generation (ING) also chooses h^* links arbitrarily to modify their costs by a randomly selected integral offset value between $-d$ and d. This offset is derived from a link-specific triangle distribution whose vertex is determined by the relative rank $r_{rel}^{\mathbf{k}}(l)$ of the respective link l. This is visualized in Figure 3. In contrast to GNG, the cost of any link has the same chance to be changed and if so, it can be increased and decreased. Like GNG, ING also favors the increase (decrease) of the cost of links with high (low) maximum link utilization $\rho_{\mathcal{S}}^{max}(\mathbf{k}, l)$, but it has more possible neighboring configurations than GNG.

3.2 Performance Comparison of the Heuristics

We study the computation time of the algorithms presented above and the quality of their results. The computation time is measured both by the actual computation time of the algorithms and by the number of evaluated link cost vectors $\mathbf{k_{new}}$; note that there is no linear mapping between these quantities since the calculation of $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k})$ may be stopped early when a preliminary result is already too large. The optimization quality is captured by the scale-up factor $\theta(\mathbf{k}) = \frac{\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{1})}{\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{k})}$ which has the following interpretation: a routing based on link costs $\mathbf{k}$ can carry the same traffic matrix scaled by factor $\theta(\mathbf{k})$ to reach the same maximum link utilization $\rho_{\mathcal{S},\mathcal{E}}^{max}(\mathbf{1})$ as the routing based on the standard hop metric $\mathbf{k} = \mathbf{1}$.

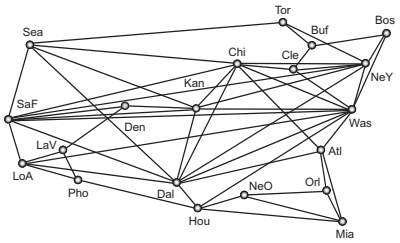
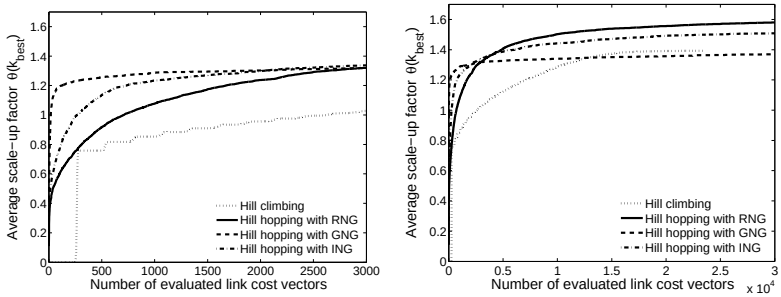


Fig. 4. The Labnet03 network consists of 20 nodes and 53 bidirectional links.

We use the Labnet03 network in our study with equal link bandwidths together with the population-based traffic matrix from [15] (cf. Figure 4). The ECMP option is used for traffic forwarding, the parameters for the heuristics are set to $k_{max} = 10$, $n_{moves}^{unsuc} = 30000$, $h = 5$, $d = 1$, $m = 1$ (for GNG), and the set of protected failure scenarios $\mathcal{S}$ comprises all single link failures.

Computation Time The heuristics improve the quality of their results incrementally, and may take very long depending on the termination criterion n_{moves}^{unsuc}. Therefore, preliminary results are already available before the program ends and we take advantage of that fact to compare the average convergence speed of 100 different optimization runs for all heuristics. The start vector $\mathbf{k_{start}}$ can be viewed as the seed for both the deterministic hill climbing algorithm and the stochastic hill hopping algorithm. For hill hopping we initialize start vectors $\mathbf{k_{start}}$ with random numbers between 1 and k_{max} while for hill climbing we use 1 and 2 with equal probability since hill climbing cannot escape from a local optimum.



(a) The first 3000 evaluations. (b) The first 30000 evaluations.

Fig. 5. Evolution of the scale-up factor $\theta(\mathbf{k})$ for different heuristics depending on the number of evaluated link cost vectors $\mathbf{k_{new}}$ and averaged over 100 runs.

Figures 5(a) and 5(b) illustrate the evolution of the scale-up factor $\theta(\mathbf{k})$ depending on the number of evaluated link cost vectors for the hill climbing and the three hill hopping variants averaged over 100 different runs. They show the scale-up factor for the first 3000 and 30000 evaluations, respectively. Due to the random initialization, the scale-up factor for the hill climbing algorithm is on average below 1 for the first 3000 evaluations, but it achieves good scale-up factors in the end. We observe the first improvement for hill climbing on average after 265 evaluations because of the chosen random initialization. As hill climbing terminates relatively early in a local optimum, the corresponding curve ends at 24000 evaluations; this has been the maximum number of evaluations in 100 runs. Hill hopping with GNG leads very fast to good results, but it is outperformed by all other algorithms on the long run. An analysis of the resulting link costs shows that most of them take either the minimum or the maximum value, i.e., this heuristic is not able to leave certain local optima. Hill hopping with ING also

yields good results quite quickly, but RNG produces better link costs after sufficiently many evaluations. Thus, sharpening the search for good candidates in the neighborhood of the current link costs **k** accelerates the convergence of the scale-up factor, but it also impedes the random discovery of excellent configurations.

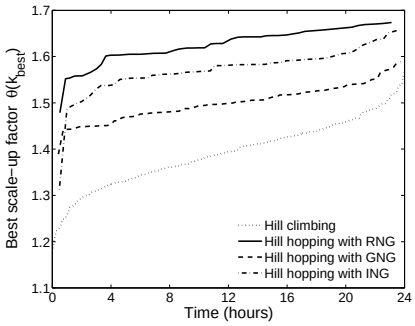


Fig. 6. Ordered scale-up factors from iterative optimization runs within 24 hours.

Quality of the Results We run the heuristics repeatedly with different seeds over 24 hours. After each termination of hill hopping, we applied an additional hill climbing to the final result to make sure that the local optimum is found. Table 1 shows that the presented algorithms run with a different frequency because they evaluate a different number of link cost vectors and some of these evaluations are stopped early.

We sort the runs according to ascending scale-up factors and present the time series of their cumulated computation times in Figure 6. It shows that the RNG variant of hill hopping leads to the best results, followed by ING, GNG, and normal hill climbing. Investigating different networks showed that the order of efficiency of the different algorithms remains the same, but the distance between the curves varies. We observed that RNG requires a low maximum link cost k_{max} to limit the search space whereas ING also works well for large k_{max}. We also tested other heuristics with similar computational requirements, e.g. the original threshold accepting (TA) algorithm [14] and simulated annealing (SA), but hill hopping leads to the best results. In addition, hill hopping has fewer parameters than TA or SA and is, therefore, simpler to apply.

Table 1. Number of optimization runs within 24 hours with the corresponding number of evaluated link cost vectors.

method	#runs	#evals/run	#evals in 24h
hill climbing	358	13719	4911386
hill hopping (GNG)	95	63079	5992505
hill hopping (ING)	58	110752	6423628
hill hopping (RNG)	48	90032	4321527

4 IP Resilience for Multilayer Networks

We first comment on multilayer resilience. Then we discuss various protection variants with different implications on the resource management which impact the objective function of the optimization problem. Finally, we compare the different protection variants.

4.1 Multilayer Resilience

As mentioned in Section 1, networks have a layered architecture as illustrated in Figure 1. Several layers can provide resilience mechanisms with backup capacity to repair broken paths. Link management or routing protocols trigger their activation, and the temporal coordination of the resilience mechanisms of different layers is an important issue that is solved, e.g., by timers. The reaction time on lower layers must be shorter than on upper layers to avoid unnecessary and repeated reroutes on upper layers. As a consequence, cable cuts are repaired by lower layer protection while the outage of

IP routers still requires IP rerouting to reestablish connectivity. As any failure can be repaired on upper layers, multilayer resilience seems a waste of resources. However, lower layer protection mechanisms are faster than IP rerouting since they switch the traffic to preestablished backup paths in case of a failure. Therefore, multilayer resilience is used in practice, but it is desirable to save backup capacity by reusing the backup capacity of the MPLS layer on the IP layer whenever possible.

4.2 Optimization of IP Routing in Multilayer Networks

We now consider different options for multilayer resilience. They differ in reaction speed and the available capacity after rerouting. The failure of IP nodes must be protected by slow IP rerouting. In contrast, IP link failures, which are more likely, can be healed by slow IP rerouting if no link layer protection exists (NoLLP), by fast 1:1 link layer protection (1:1LLP), or by very fast 1+1 link layer protection (1+1LLP). We talk about low, medium, and high service availability (LSA, MSA, HSA) if there is no explicit backup capacity, if the capacity suffices to carry the backup traffic from single link failures, or from single link and router failures. In the following, we discuss different link protection alternatives with different requirements for service availability.

No Link Layer Protection with Low, Medium, and High Service Availability (NLLP-LSA, NLLP-MSA, NLLP-HSA) As failures are protected only by IP rerouting, the full capacity is available for the IP layer and Equation (3) can be used as objective function for the routing optimization. The service availability impacts the set of protected failure scenarios such that $\mathcal{S}$ contains the failure-free scenario only, all single link failures, or all single link and router failures for NoLLP-LSA, NoLLP-MSA, and NoLLP-HSA, respectively.

Link Layer Protection with Medium Service Availability (LLP-MSA) In the presence of 1+1 or 1:1 link layer protection, IP routing can be optimized for the failure-free case $\mathcal{S} = \emptyset$ since link failures are completely covered by LLP and node failures do not need to be protected. We assume that backup capacity sharing is not possible and that LLP consumes 50% of the link layer capacity. Therefore, the utilization values of the IP layer capacity are twice as large as in a network with NoLLP if the same link layer capacity is available. To get meaningful comparative results, we change Equation (3) to

$$\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k}) = 2 \cdot \max_{l \in \mathcal{E}} \left(\rho^{max}_{\mathcal{S}}(\mathbf{k}, l) \right). \tag{4}$$

1:1 Link Layer Protection with High Service Availability (1:1LLP-HSA) With 1:1LLP, the link layer provides a primary link and a backup link to the IP layer. If the primary link fails, the traffic is automatically redirected to the backup link, otherwise the backup link can carry extra traffic. Thus, only half the capacity can be used for premium traffic in failure-free scenarios. We account for this fact by calculating the utilization of the primary capacity which is twice the utilization of the overall link capacity. As we protect all single failures in our study, all links work when a router fails such that the capacity of the backup links can be reused in this case. Thus, the full link capacity is available for rerouting due to node failures. To capture these side conditions, we substitute Equation (3) for the optimization of resilient IP routing by

$$\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k}) = \max \left(2 \cdot \max_{l \in \mathcal{E}} \rho(\mathbf{k}, l, \emptyset), \max_{\{(l,s): l \in \mathcal{E}, s \in \mathcal{S} \wedge s \neq \emptyset\}} \rho(\mathbf{k}, l, s) \right) \tag{5}$$

where the set of protected failure scenarios $\mathcal{S}$ comprises all single router failures.

1+1 Link Layer Protection with High Service Availability (1+1LLP-HSA) With 1+1LLP, traffic is simultaneously carried over primary and backup paths such that the reaction time is very short if a failure occurs. Hence, the backup capacity can never be reused on the IP layer, and only half of the link layer capacity is available for IP traffic. Therefore, Equation (4) applies instead of Equation (3) for the optimization of resilient IP routing with $\mathcal{S}$ being the set of all single router failures.

Related Aspects We briefly mention additional issues that have not been taken into account by the above scenarios and may be for further study.

Shared Protection on Lower Layers The above scenarios assumed that on the lower layer, the capacity of a backup path is fully dedicated to a single primary path. When shared protection is allowed, the same capacity carries backup traffic from different primary paths in different failure scenarios. As a consequence, significantly more than 50% of the link capacity can be used to carry protected IP traffic on the IP layer [15]. The authors of [16] have shown that single link failures can be protected more efficiently by plain WDM protection than by plain IP restoration if backup capacity sharing is allowed for both options.

Shared Risk Groups (SRGs) For simplicity reasons, we consider only single link or router failures. However, multi-failures may also occur due to simultaneous uncorrelated failures or due to correlated failures of so-called shared risk groups (SRGs). The simplest form of a shared risk link group (SRLG) is the failure of a router which entails the simultaneous failure of all its adjacent links. More complex SRGs occur, e.g., due to the failure of unprotected lower layer equipment. General SRGs can be integrated in our optimization approach by simply including them into the set of protected failure scenarios $\mathcal{S}$, but in practice, the difficulty is mostly the missing knowledge about them.

4.3 Performance Comparison

We compare the bandwidth efficiency of the multilayer resilience scenarios with and without routing optimization. We calculate the maximum link utilization $\rho^{max}_{\mathcal{S},\mathcal{E}}(X,\mathbf{k})$ for the hop count metric ($\mathbf{k}=\mathbf{1}$) and for optimized link cost $\mathbf{k_{best}}$ for each multilayer resilience scenario X. As the maximum utilization value $\rho^{max}_{\mathcal{S},\mathcal{E}}(X,\mathbf{k})$ is the largest for unoptimized routing in the 1+1LLP-HSA scenario, we use $\rho^{max}_{\mathcal{S},\mathcal{E}}(\text{1+1LLP-HSA},\mathbf{1})$ as the base for the relative scale-up factor $\eta(X,\mathbf{k_{best}}) = \frac{\rho^{max}_{\mathcal{S},\mathcal{E}}(\text{1+1LLP-HSA},\mathbf{1})}{\rho^{max}_{\mathcal{S},\mathcal{E}}(X,\mathbf{k_{best}})}$.

Table 2 presents results from the Labnet03 (cf. Figure 4) both for the heterogeneous traffic matrix used above and for a homogeneous traffic matrix. The scale-up factors $\eta(X,1)$ and $\eta(X,\mathbf{k_{best}})$ illustrate the impact of the multilayer scenario X. They quantify how much more traffic can be carried with X compared to 1+1LLP-HSA. Obviously, most traffic can be transported with NoLLP-LSA, followed by NoLLP-MSA, NoLLP-HSA, LLP-MSA, 1:1LLP-HSA, and 1+1LLP-HSA. In the presence of a heterogeneous traffic matrix, the protection of link and router failures requires more backup capacity than the protection of only link failures when no LLP is used. In contrast, in the presence of the homogeneous traffic matrix NoLLP-HSA and NoLLP-MSA need about the same backup resources and 1:1LLP-HSA is as efficient as LLP-MSA, i.e., the protection of additional router failures does not cost extra resources. Backup capacity sharing between the link and the network layer makes 1:1LLP-HSA 27%-56% more efficient than 1+1LLP-HSA. Hence, if the reaction time of IP rerouting is not acceptable, 1:1LLP-HSA may be preferred as the more efficient alternative to 1+1LLP-HSA.

However, 1+1LLP-HSA reacts faster than 1:1LLP-HSA if links fail and may be applied when very fast resilience is needed.

Table 2. Scale-up factors for optimized and unoptimized IP routing in different multilayer resilience scenarios.

resilience scenario X	w/o IP opt $\eta(X, \mathbf{1})$	with IP opt $\eta(X, \mathbf{k_{best}})$	$\theta(X, \mathbf{k_{best}})$
hetero TM			
NoLLP-LSA	3.13	4.76	1.52
NoLLP-MSA	2.25	3.41	1.52
NoLLP-HSA	2.00	2.63	1.32
LLP-MSA	1.56	2.38	1.52
1:1LLP-HSA	1.56	1.97	1.26
1+1LLP-HSA	1.00	1.32	1.32
homo TM			
NoLLP-LSA	2.54	4.60	1.81
NoLLP-MSA	1.93	3.27	1.69
NoLLP-HSA	1.93	3.21	1.66
LLP-MSA	1.27	2.30	1.81
1:1LLP-HSA	1.27	2.29	1.80
1+1LLP-HSA	1.00	1.68	1.68

The scale-up factor of Section 3 can be alternatively calculated by $\theta(X, \mathbf{k_{best}}) = \frac{\eta(X, \mathbf{k_{best}})}{\eta(X, \mathbf{1})}$ and shows the benefit of routing optimization for each scenario X. Routing optimization improves the resource efficiency by 26%-52% in case of the heterogeneous traffic matrix and by 66%-81% in case of a homogenous traffic matrix. It is so powerful that more traffic can be carried with optimized NoLLP-HSA than with unoptimized NoLLP-LSA in case of the heterogeneous traffic matrix. Thus, with routing optimization, resilience can be achieved without additional bandwidth.

5 Summary and Conclusion

As overload in networks is mostly caused by redirected traffic due to network failures [1], administrative IP link costs should be set in such a way that the maximum utilization $\rho^{max}_{\mathcal{S},\mathcal{E}}(\mathbf{k})$ of the links is low both under failure-free conditions and in likely failure scenarios. We presented the hill climbing and the hill hopping algorithms with different neighborhood generation strategies for the optimization of resilient IP routing. A comparison showed that some of them converge faster, but others lead to better optimization results. The presented methodology for the performance comparison is general and can be applied to other heuristic approaches.

Different levels of service availability and multilayer resilience change the side conditions for the optimization because different failures need to be protected and depending on the technology, some of the physical layer capacity is dedicated to lower layers or can be shared among layers. Our results showed that with routing optimization 32% to 81% more traffic can be carried in our test network while keeping the same maximum utilization as without routing optimization. The exact values depend both on the required level of service availability, the multilayer resilience option, and the traffic matrix. Furthermore, the network itself has a large impact which has not been documented in this paper. Routing optimization turned out to be so powerful that protection of link and node failures leads in some settings to lower maximum link utilizations than unoptimized routing under failure-free conditions.

References

1. Iyer, S., Bhattacharyya, S., Taft, N., Diot, C.: An Approach to Alleviate Link Overload as Observed on an IP Backbone. In: IEEE Infocom, San Francisco, CA (April 2003)
2. Pióro, M., Szentesi, Á., Harmatos, J., Jüttner, A., Gajowniczek, P., Kozdrowski, S.: On Open Shortest Path First Related Network Optimisation Problems. Performance Evaluation **48** (2002) 201 – 223
3. Fortz, B., Thorup, M.: Internet Traffic Engineering by Optimizing OSPF Weights. In: IEEE Infocom, Tel-Aviv, Israel (2000) 519–528
4. Iannaccone, G., Chuah, C.N., Bhattacharyya, S., Diot, C.: Feasibility of IP Restoration in a Tier-1 Backbone. IEEE Network Magazine (Special Issue on Protection, Restoration and Disaster Recovery) (March 2004)
5. Fortz, B., Thorup, M.: Robust Optimization of OSPF/IS-IS Weights. In: International Network Optimization Conference (INOC), Paris, France (October 2003) 225–230
6. Yuan, D.: A Bi-Criteria Optimization Approach for Robust OSPF Routing. In: 3^{rd}IEEE Workshop on IP Operations and Management (IPOM), Kansas City, MO (October 2003) 91 – 98
7. Nucci, A., Schroeder, B., Bhattacharyya, S., Taft, N., Diot, C.: IGP Link Weight Assignment for Transient Link Failures. In: 18^{th} International Teletraffic Congress (ITC), Berlin (September 2003)
8. Sridharan, A., Guerin, R.: Making IGP Routing Robust to Link Failures. In: IFIP-TC6 Networking Conference (Networking), Ontario, Canada (May 2005)
9. Demeester, P., Gryseels, M., Autenrieth, A., Brianza, C., Castagna, L., Signorelli, G., Clemente, R., Ravera, M., Jajszczyk, A., Janukowicz, D., Doorselaere, K.V., Harada, Y.: Resilience in Multilayer Networks. IEEE Communications Magazine (August 1999) 70 – 76
10. Oran, D.: RFC1142: OSI IS-IS Intra-Domain Routing Protocol (February 1990)
11. Rai, S., Mukherjee, B., Deshpande, O.: IP Resilience within an Autonomous System: Current Approaches, Challenges, and Future Directions. IEEE Communications Magazine (October 2005) 142–149
12. Shand, M., Bryant, S.: IP Fast Reroute Framework. `http://www.ietf.org/internet-drafts/draft-ietf-rtgwg-ipfrr-framework-06.txt` (October 2006)
13. Fortz, B., Thorup, M.: Optimizing OSPF/IS-IS Weights in a Changing World. IEEE Journal on Selected Areas in Communications **20** (May 2002) 756 – 767
14. Dueck, G., Scheuer, T.: Threshold Accepting; a General Purpose Optimization Algorithm. Journal of Computational Physics **90** (1990) 161–175
15. Menth, M.: Efficient Admission Control and Routing in Resilient Communication Networks. PhD thesis, University of Würzburg, Faculty of Computer Science, Am Hubland (July 2004)
16. Sahasrabuddhe, L., Ramamurthy, S., Mukherjee, B.: Fault Tolerance in IP-Over-WDM Networking: WDM Protection vs. IP Restoration. IEEE Journal on Selected Areas in Communications (Special Issue on WDM-Based Network Architectures) **20**(1) (Jan. 2002) 21–33