Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

ANDREAS BLEY

# A Lagrangian Approach for Integrated Network Design and Routing in IP Networks

URL: http://www.zib.de/PaperWeb/abstracts/ZR-03-29

ZIB-Report 03-29 ( 2003)

# A Lagrangian Approach for Integrated Network Design and Routing in IP Networks*

Andreas Bley
Konrad-Zuse-Zentrum
Address : Takustr. 7, 14195 Berlin, Germany
Fax: +49-30-84185269
Email: bley@zib.de

**Abstract**

We consider the problem of designing a network that employs a non-bifurcated shortest path routing protocol. The network's nodes and the set of potential links are given together with a set of forecasted end-to-end traffic demands. All relevant hardware components installable at links or nodes are considered. The goal is to simultaneously choose the network's topology, to decide which hardware components to install on which links and nodes, and to find appropriate routing weights such that the overall network cost is minimized.

In this paper, we present a mathematical optimization model for this problem and an algorithmic solution approach based on a Lagrangian relaxation. Computational results achieved with this approach for several real-world network planning problems are reported.

**Keywords:** Network Planning, Routing, IP Networks, Lagrangian relaxation

## 1 Introduction

Most routing protocols in today's data networks are based on shortest path routing. Open Shortest Path First (OSPF), for example, is the most commonly used intra-domain routing protocol in the Internet at the moment. Shortest path routing protocols are based on administrative routing weights that are assigned to the links of the network. These weights determine the routing paths: Data packages are forwarded from their source to their destination on a shortest path with respect to the assigned weights. The fact that the routing paths and traffic flows can be (re-)configured only to some limited degree and only indirectly by modifying the administrative routing weights makes network design and traffic engineering in these networks often very difficult.

In this paper, we study the problem of designing a cost minimal network that uses a non-bifurcated shortest path routing protocol. We are given network's node locations, the set of all potential links between these nodes, and a set of forecasted end-to-end traffic demands. Furthermore, for each of the nodes and links we are given the list of hardware components that can be installed as well as a number of constraints that a hardware installation must satisfy. Such components may be router racks, interface cards, or link configurations corresponding to certain capacity levels, for example. The given constraints describe which combinations of components are locally and globally valid, i.e., can be 'plugged together' without violating technical or operational side-constraints. The task of the integrated network design and routing problem is to decide simultaneously which components to install at the nodes and links (thereby also decide the network's topology and node and link capacities) and to

*Partially supported by the Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMBF) and by the DFG research center "Mathematics for key technologies" (FZT 86) in Berlin."

provide the routing weights which determine the traffic flows such that the capacities are not exceeded. The objective is to minimize the overall network cost, which is the sum of the costs of all installed components.

Extensions that allow to split the traffic among several shortest paths are not considered in this paper. These are mostly non-standardized and proprietary. We restrict our attention to the case of non-bifurcated shortest path routing, i.e., all data packages between two nodes must be sent on the same path. Furthermore, we consider only the case where different routing weights may be chosen for the two directions of a link and, hence, asymmetric routing is possible.

Over the last decade, various mathematical models and solution techniques for this and related problems have been developed. Ben-Ameur and Gourdin [BAG00] studied the structural properties of undirected shortest path routings and developed linear programming formulations for the corresponding inverse shortest path problem, where the task is to find routing weights that induce a given set of paths. They also addressed the issue of finding small integer weights, which is a crucial issue for some shortest path routing protocols. A Lagrangian relaxation approach similar to ours was proposed by Lin and Wang [LW93] for a traffic engineering problem in networks with shortest path routing. Given a capacitated network and some end-to-end traffic demands, the task is to find routing weights such that the maximum congestion of the links is minimized. For a similar problem, where traffic splitting is allowed and the objective is to minimize the sum over all links of a barrier-like penalty function of the single link's congestion, a number of genetic and local search algorithms have been developed by various groups of authors [ERP01, FT00, FGL+00]. Integer programming formulations for non-bifurcated shortest path routing problems have been proposed by Berry et. al. [BKSTG00] for traffic engineering problems and by Holmberg and Yuan [HY01] and Prytz [Pry02] for network design problems with discrete link capacities. Bley et al. [BGW98, BK02] studied the network design problem with additional survivability constraints and presented mixed-integer programming approaches for traffic engineering and network design problems with non-bifurcated shortest path routing and a modular link and node hardware model.

The remainder of this paper is organized as follows. In Section 2, we present a mathematical model of the integrated network design and shortest path routing problem. The Lagrangian relaxation based solution approach is described in Section 3. Finally in Section 4, we report on computational results for a number of instances that are based on real-world planning problems.

# 2 Problem and model

## 2.1 Hardware configuration

Let $V$ denote the set of all node locations and $E$ the set of all potentially installable links. These form the undirected supply graph $G = (V, E)$.

At each node and each link that is used in the network, appropriate node hardware or link capacities must be installed. In practice, these systems are combinations of various devices or transmission systems. For example, an IP router consists of a router rack and a number of interface cards. Of course, these devices cannot be combined arbitrarily. A number of technical constraints must be respected. It depends on the chosen router rack type what types of and how many router interface cards can be installed. If a certain capacity is installed on a link, the corresponding interfaces must be provided by some router interface card at the two end-nodes. In the planning of IP-over-SDH or IP-over-WDM networks, the SDH or WDM equipment usually has to be considered, too.

We model the different node hardware devices and the link capacities as abstract node and link components. For each node $v \in V$, we are given the set of all components $C(v)$ that are available at $v$. The set of possible link capacities for $e \in E$ is $C(e)$. Local technical or administrative constraints, like the usage of router card slots at a node or the number of link capacities that can be installed in parallel on a link, are modeled by node and link specific resources $r \in R(v)$, for $v \in V$, and $r \in R(e)$, $e \in E$, respectively. The contribution of component $c$ to a resource $r$ at some node $v$ is denoted by $\alpha_v^{c;r}$. If $\alpha_v^{c;r} > 0$, then installing one component $c$ provides $\alpha_v^{c;r}$ units of $r$ at $v$, if $\alpha_v^{c,r} < 0$, then $|\alpha_v^{c,r}|$ units are consumed. For example, installing a router rack at a node provides a number of empty router card slots while installing a link capacity consumes the corresponding link interfaces at both end-nodes. Analogous, the contribution of a component $c$ to a link specific resource $r$ at $e \in E$ is denoted by $\alpha_e^{c,r}$. With an appropriate choice of components and resources, most technical constraints related to the modular structure of the hardware can be expressed. At each node and each link, the consumption of a resource must not exceed the provision by the installed components. With the help of global components $C(G)$, which contribute to the resources at all nodes and links in the network, and global resources $R(G)$, which impose global constraints on all the components installed in the network, we can also model global aspects of the network design problem. This includes, for example, global budgets, network reconfiguration restrictions, or rebates for purchasing bundles of components. Other local or global side constraints, like the requirement to provide some additional link interfaces at certain nodes for peering, can be modeled with artificial components.

The number of installations of node, edge, and global components is modeled by integer variables $z_v^c \in \mathbb{N}$, $z_e^c \in \mathbb{N}$, and $z_G^c \in \mathbb{N}$, respectively. Lower and upper bound on the number of installations of a node component $c$ at a node $v$ are expressed by $z_{v,min}^c$ and $z_{v,max}^c$, respectively, and analogous for edge and global components. The problem of choosing a valid hardware installation (and, thereby, also the network's topology) can be formulated as follows:

$$\begin{aligned}
\sum_{e\in\delta(v)}\sum_{c\in C(e)} \alpha_e^{c,r} z_e^c + \sum_{c\in C(v)} \alpha_v^{c,r} z_v^c &\geq 0 \quad \text{for all } v\in V,\ r\in R(v), && (1)\\
\sum_{c\in C(e)} \alpha_e^{c,r} z_e^c + \sum_{c\in C(u)} \alpha_u^{c,r} z_u^c + \sum_{c\in C(v)} \alpha_v^{c,r} z_v^c &\geq 0 \quad \text{for all } uv=e\in E,\ r\in R(e), && (2)\\
\sum_{e\in E}\sum_{c\in C(e)} \alpha_e^{c,r} z_e^c + \sum_{v\in V}\sum_{c\in C(v)} \alpha_v^{c,r} z_v^c + \sum_{c\in C(G)} \alpha_G^{c,r} z_G^c &\geq 0 \quad \text{for all } r\in R(G), && (3)\\
\mathbf{z}_{min} \leq \mathbf{z} &\leq \mathbf{z}_{max}\,. && (4)
\end{aligned}$$

Inequalities (1) state that at each node and for each node resource the amount consumed must not exceed the amount provided of that resource by the installed components. The analogous constraint is expressed for link and global resources in (2) and (3), respectively. The lower and upper bounds on the number of installations of components are expressed in (4). Let $\mathcal{C} := \{(v,c) : v \in V,\ c \in C(v)\} \cup \{(e,c) : e \in E,\ c \in C(e)\} \cup \{c : c \in C(G)\}$ be the component variable space and define

$$Z := \{\mathbf{z} \in \mathbb{N}^{\mathcal{C}} : \mathbf{z} \text{ satisfies (1)–(4)}\}.$$

## 2.2 Routing

For the Lagrangian solution approach presented in the next section, variables that explicitly describe the routing paths of the demands are not needed. The following model of the non-bifurcated shortest path routing implicitly describes the relation between the routing weights and the resulting link flows.

For each pair of nodes $u, v \in V$, let $d^{u,v} \in \mathbb{R}$, $d^{u,v} \geq 0$, denote the directed traffic demand from $u$ to $v$. For each undirected link $uv \in E$, we denote by $(u,v)$ and $(v,u)$ its two associated directed arcs and set $A := \{(u,v),\ (v,u) : uv \in E\}$. We assume that components installed on an undirected link $uv \in E$ provide the same routing capacity for both directions of the link. The resource corresponding to the provision of this routing capacity is denoted by *cap*, $cap \in R(e)$ for all $e \in E$. The routing weights assigned to the arcs $(u,v) \in A$ are modeled by the variables $w_{(u,v)} \in \mathbb{N}$. The traffic flows resulting from the chosen routing weights for the given demands are expressed by the variables $f_{(u,v)} \in \mathbb{R}_+$, $(u,v) \in A$. We define

$$F := \left\{ (\mathbf{f},\mathbf{w}) \in (\mathbb{R}^A, \mathbb{N}^E) : \begin{array}{l} \mathbf{w} \text{ induces unique shortest paths between all node pairs } u,v \in V,\ u \neq v, \text{ and} \\ \mathbf{f} \text{ is induced flow of the corresponding shortest path routing of the demands } \mathbf{d} \end{array} \right\}.$$

With this notation, the problem of finding a shortest path routing can be formulated as follows:

$$(\mathbf{f}, \mathbf{w}) \quad \in \quad F\,, \tag{5}$$
$$f_{(u,v)} \quad \leq \quad \sum_{c \in C(uv)} \alpha_{uv}^{c,cap} z_{uv}^{c} \quad \text{for all } (u,v) \in A. \tag{6}$$

Constraint (5), which involves the implicitely defined set $F$, ensures that $\mathbf{w}$ induces unique shortest paths for all demands and that $\mathbf{f}$ is the corresponding flow on the directed links. Inequality (6) guarantees that this flow does not exceed the provided link capacities.

Note that, in order to guarantee non-ambiguous non-bifurcated shortest path routing, the routing weights are required to induce unique shortest paths between all node pairs $u, v \in V$. If the routing weights do not satisfy this requirement but non-bifurcated routing is enforced, it solely depends on the implementation of the routing protocol in the real network, which of the shortest paths is chosen in case of ambiguity. As the administrator may loose some control over the routing and capacities might no longer be adequate for the real traffic flows, such routing weights should be strictly avoided. If the implemented routing protocol resolves ambiguities in a stable and deterministic manner, then there also exist non-ambiguous routing weights that induce the same routing paths.

Another issue is that the weights in all real routing protocols are bounded, in OSPF between 0 and 65535, for example. Nevertheless, even for large networks it is possible to find small integer routing weights that induce the same shortest paths like some given other (even non-integer valued) routing weights do. If one is not interested in minimizing the maximum or the sum of the weights but only in finding integer weights within the given bounds, a simple scaling-and-rounding procedure or solving the integer inverse shortest path problem for the path set induced by the given weights are computationally admissible for real-world size networks.

### 2.3 Cost minimization

The objective of the network design problem is to minimize the total network cost. If we let *cost* $\in R(G)$ denote the global cost resource, the objective function can be formulated as follows:

$$\min \quad \mathbf{k}^T \mathbf{z} := \sum_{e \in E} \sum_{c \in C(e)} \alpha_e^{c,cost} z_e^c + \sum_{v \in V} \sum_{c \in C(v)} \alpha_v^{c,cost} z_v^c + \sum_{c \in C(G)} \alpha_G^{c,cost} z_G^c. \tag{7}$$

Of course, any other objective that is linear in the components can be formulated via an appropriate resource.

## 3 A Lagrangian solution algorithm

In this section, we present our solution approach for the integrated network design and shortest path routing problem and discuss some implementation details.

### 3.1 The Lagrangian relaxation

The complete model (1)–(7) can be written as

$$\left\{ \begin{array}{ll} & \min\ \mathbf{k}^T \mathbf{z} \\ \text{s.t.} & f_{(u,v)} \leq \displaystyle\sum_{c \in C(uv)} \alpha_{uv}^{c,cap} z_{uv}^{c} \qquad \text{for all } (u,v) \in A \\ & \mathbf{z} \in Z \\ & (\mathbf{f}, \mathbf{w}) \in F \end{array} \right\} \tag{8}$$

We relax the capacity constraints and denote the corresponding Lagrangian dual multipliers by $\mu_{(u,v)} \geq 0$ for all $(u,v) \in A$. The resulting Lagrangian function is

$$
\begin{aligned}
L(\mu) &= \min \left\{ \mathbf{k}^T \mathbf{z} - \sum_{(u,v)\in A} \sum_{c \in C(uv)} \mu_{(u,v)} \alpha_{uv}^{c,cap} z_{uv}^c + \sum_{(u,v)\in A} \mu_{(u,v)} f_{(u,v)} \;:\; \mathbf{z} \in Z \,,\, (\mathbf{f}, \mathbf{w}) \in F \right\} \\
&= \underbrace{\min \left\{ \mathbf{k}^T \mathbf{z} - \sum_{(u,v)\in A} \sum_{c \in C(uv)} \mu_{(u,v)} \alpha_{uv}^{c,cap} z_{uv}^c \;:\; \mathbf{z} \in Z \right\}}_{L^Z(\mu)} + \underbrace{\min \left\{ \sum_{(u,v)\in A} \mu_{(u,v)} f_{(u,v)} \;:\; (\mathbf{f}, \mathbf{w}) \in F \right\}}_{L^F(\mu)\,.}
\end{aligned}
\tag{9}
$$

It is well known that, for each dual vector $\mu \in \mathbb{R}_+^A$, the value $L(\mu)$ is a lower bound for the optimal value of the original problem (8). Hence,

$$
L^* := \max_{\mu \in \mathbb{R}_+^A} L(\mu) \leq \mathbf{k}\mathbf{z}^*,
$$

where $(\mathbf{z}^*, \mathbf{f}^*, \mathbf{w}^*)$ is an optimal solution of (8). As there are only finitely many different (basic) solutions $\mathbf{z}$ and $\mathbf{f}$ with $\mathbf{z} \in Z$ and $(\mathbf{f}, \mathbf{w}) \in F$, both functions $L^Z(\mu)$ and $L^F(\mu)$ are concave in $\mu$. Hence, $-L(\mu)$ is convex and the problem of finding the optimal dual multipliers $\mu$ can be solved by a general convex function optimization algorithm.

Note that exactly the same Lagrangian function $L(\mu)$ and the same lower bound $L^*$ is obtained, if the analogous network design problem with general multicommodity flow routing instead of non-bifurcated shortest path routing is relaxed. Hence, we cannot expect to obtain very tight lower bounds with this approach.

Nevertheless, this Lagrangian approach is attractive for practical computations. One reason is that the Lagrangian function $L(\mu)$ decomposes into the sum of two functions $L^Z(\mu)$ and $L^F(\mu)$, both of which can be evaluated efficiently for real-world size networks.

Evaluating the first function $L^Z(\mu)$ corresponds to the problem of finding a valid hardware installation that minimizes a linear objective function. Although this problem may be $\mathcal{NP}$-hard in general, its integer programming formulation can be solved very efficiently by state-of-the-art integer programming solvers for real-world problems. Note that this formulation contains only the variables $\mathbf{z}$ and the inequalities (1)–(4). The traffic demands and flows only affect the objective function coefficients via the Lagrangian dual multipliers. For simple hardware models, this problem decomposes even further. For example, if only one capacity may be chosen for each link and no node or global components or constraints are considered, choosing for each edge the minimum cost capacity yields a cost minimal overall hardware installation.

The optimization problem associated with evaluating the second function $L^F(\mu)$ can be solved by any shortest path algorithm. The task is to find a non-bifurcated shortest path routing that minimizes the total flow costs for $\mu$. It is not hard to see that choosing for each pair of nodes the shortest path with respect to $\mu$ yields the optimal solution. Ties between equally long shortest paths can be broken by using an arbitrary numbering of the nodes or links as a secondary length function (or perturbing $\mu$ accordingly, see [BGW98]). This guarantees the existence of some integer routing weights that induce the same set of shortest paths as $\mu$.

The second reason for this Lagrangian relaxation approach being computationally interesting is the possibility to include other heuristics. After each iteration of the convex optimization algorithm, the current duals $\mu$ can be easily interpreted as routing weights, as in the evaluation of $L^F(\mu)$. In practice, these weights seem to provide good starting points for other heuristics that are based on evaluation of routing weights. From this perspective, our Lagrangian relaxation approach also can be seen as a primal heuristic that modifies the current routing weights according to the dual information and, as a byproduct, produces a lower bound for the optimum solution value.

### 3.2 Implementation

The presented solution approach is implemented as part of the DISCNET network optimization library [ate03]. The data structures and algorithms are based on the standard C++ library and LEDA [Alg03]. The CONICBUNDLE algorithm of Helmberg [HK02, HR00] is used to solve the convex optimization problem of finding the optimal Lagrangian duals $\mu^*$. Both functions $L^Z(\mu)$ and $L^F(\mu)$ have an independent bundle of subgradients. In order to keep the number of function evaluations small, the number of directed links in the network is used as bundle-size for both functions. The integer programming problem corresponding to the evaluation of $L^Z(\mu)$ and the integer inverse shortest path problem to compute small integer routing weights when post-processing solutions are solved by CPLEX [ILO02]. For shortest path computations, we apply a variant of Dijkstra's algorithm that breaks ties between equally long shortest paths based on some numbering of the edges.

The formulation of the hardware installation subproblem $Z$ was tightened by adding band and strengthened metric inequalities [Wes00] associated with single-node and two-node cuts in the graph. These inequalities cut off many hardware installations that cannot even accommodate a fractional multicommodity flow routing of the given demands. Adding these inequalities speeds up the convergence of the algorithm and improves the lower bound $L^*$, but the evaluation time for $L^Z(\mu)$ is increased, too.

Two types of primal heuristics were used in the computations reported in the following section. The first heuristic was applied for all but the GWIN instances. It first computes the shortest path routing for the current Lagrangian duals interpreted as routing weights. Then it computes a cost-minimal hardware installation that can accommodate the resulting link flows, by solving an integer linear program. This heuristic performs well if for almost any flow of the given demands in the underlying graph there exists such a hardware installation. This is usually the case if large enough capacities are available and no constraints restrict the topology of feasible hardware configurations. If there are tight budget constraints, small node degree bounds, or very restrictive hardware reconfiguration constraints, this heuristic often fails to find feasible solutions.

The second heuristic tries to cope with these difficulties. It first computes an initial topology by solving the integer program associated with the hardware installation problem $L^Z(\mu)$ for the current duals. It sets the routing weights for those arcs contained in the initial topology to the current Lagrangian duals and for all other arcs to some large value and then continues as the first heuristic, computing a shortest path routing for these weights and a hardware installation. With this strategy, those arcs not in the initial topology are avoided by the shortest path routing, if possible. Thus, the resulting flows can often be accommodated by another feasible hardware installation. This heuristic was used for the GWIN instances, which have a tight bound on the number of installations of each link capacity type.

## 4 Computational results

In this section, we report on computational results for ten real-world based instances. The three GWIN instances originate from planning problems of the German Research Network [DFN]. The traffic demands were obtained by end-to-end measurements in the real network. For each link, one of several capacities can be chosen, but, for each capacity type, the number of installations is restricted. In consequence, only sparse subnetworks of the given supply graph are feasible. Both SDH and IP equipment is considered at the nodes. The LINK and ALL instances stem from real-world SDH and WDM network design problems that we encountered in other projects with industry. We assumed that the traffic demands must be routed on non-bifurcated shortest paths, although this is not required in the original applications. In the LINK instances only link capacities are considered, while in the ALL instances we consider all hardware components and restrictions of the original application.

The characteristic properties of the ten instances and our computational results are listed in Table 1. In order to evaluate the practical usefulness and the theoretical strength of the algorithm, we not only report the best solution, final lower bound, total time (including heuristics), and total number of dual function evaluations of the Lagrangian algorithm, but also the time when the algorithm found the best solution and an alternative

| | GWIN | | | LINK | | | ALL | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | G1 | G2 | G3 | L1 | L2 | L3 | A1 | A2 | A3 | A4 |
| Nodes | 11 | 11 | 11 | 15 | 14 | 27 | 24 | 22 | 36 | 19 |
| Links | 47 | 47 | 47 | 22 | 21 | 51 | 55 | 34 | 122 | 148 |
| Demands | 110 | 110 | 110 | 210 | 182 | 571 | 276 | 231 | 630 | 171 |
| Avg. capacities/Link | 3 | 4 | 2 | 7 | 7 | 7 | 12 | 6 | 5 | 5 |
| Avg. comps./Node | 6 | 8 | 4 | 0 | 0 | 0 | 12.2 | 9.2 | 6 | 4.1 |
| Best Solution | 108.7 | 112.3 | *100.0 | 107.0 | 109.7 | 163.7 | 106.2 | 102.4 | 110.0 | 104.8 |
| Lower Bound | 65.1 | 77.3 | 60.6 | 75.8 | 72.3 | 100.0 | *100.0 | 97.5 | 76.1 | 88.0 |
| Branch&Cut | 100.0 | *100.0 | *100.0 | *100.0 | *100.0 | 85.8 | *100.0 | *100.0 | 100.0 | 100.0 |
| Total time | 01:02 | 0:32 | 0:41 | 1:52 | 30:00 | 30:00 | 00:01 | 00:01 | 30:00 | 30:00 |
| Total evals. | 156 | 168 | 171 | 69 | 74 | 847 | 2 | 8 | 75 | 163 |
| Time best soluon | 0:07 | 0:02 | 0:06 | 00:39 | 1:18 | 6:39 | 00:01 | 00:01 | 23:45 | 2:09 |

Table 1: Test instances and computational results (times in min:sec, optimal solutions marked by * )

lower bound obtained by a branch-and-cut algorithm ([BK02]). The cost values are scaled such that the best of the two lower bound values equals 100. The computations were performed on a 3 GHz Pentium4 PC running Linux. For each instance, both the Lagrange algorithm's (including heuristics) and the branch-and-cut algorithm's computation times were limited to 30 minutes. The Lagrange algorithm terminated when the optimum Lagrangian duals were determined with a relative precision of $10^{-4}$.

From a practical viewpoint, the main advantage of the Lagrangian algorithm compared to the branch-and-cut approach is its better scalability. Even for the larger instances, the Lagrangian algorithm produces resonably good solutions very quickly and with little memory consumption. Even for instance A3, where it took 23 minutes to find the best solution, a feasible solution was found immediately and a solution with cost 111.5 in less than 5 minutes. The branch-and-cut algorithm does not scale well with the problem size, because its underlying linear programming formulation contains not only the hardware component variables but also a huge number of flow variables to model the chosen routing paths. On the other hand, in most cases the lower bound obtained by the Lagrangian approach is less than the bound obtained with the branch-and-cut algorithm. Especially for instances with restrictions on the feasible network topologies, like the GWIN instances, the difference is large. In these instances, the capacity constraints do not play the central role in linking the flows to the hardware installation. Also finding good solutions is harder in general for such instances. For larger networks with less topology-restrictive constraints, the flows and hardware installation can be easier 'adjusted' to each other via the Lagrangian duals and both the lower bounds and the solutions computed by the Lagrangian approach are reasonably good.

# 5 Conclusions

We proposed a Lagrangian relaxation approach to solve the problem of designing a network that uses a non-bifurcated shortest path routing protocol. Albeit this approach produces worse lower bounds than a branch-and-cut algorithm, our computational results demonstrate that this approach is useful to compute good solutions for real-world problems in a very short time, especially for large networks.

# References

[Alg03] Algorithmic Solutions Software GmbH, Schützenstr. 3-5, D- 66123 Saarbrücken, Germany, LEDA—*Library of Efficient Data types and Algorithms*, 1998–2003, Information available at `http://www.algorithmic-solutions.com`.

[ate03] atesio GmbH, Rubensstr. 126, D-12157 Berlin, Germany, DISCNET, 2000–2003, Information available at http://www.atesio.de.

[BAG00] W. Ben-Ameur and E. Gourdin, *Internet routing and related topology issues*, submitted to SIAM J. Discrete Mathematics, 2000.

[BGW98] A. Bley, M. Grötschel, and R. Wessäly, *Design of broadband virtual private networks: Model and heuristics for the B-WiN*, Robust Communication Networks: Interconnection and Survivability (N. Dean, D. F. Hsu, and R. Ravi, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 53, AMS, 1998, pp. 1–16.

[BK02] A. Bley and T. Koch, *Integer programming approaches to access and backbone IP-network planning*, Tech. Report ZR 02-41, Konrad-Zuse Zentrum für Informationstechnik Berlin (ZIB), 2002.

[BKSTG00] L. Berry, S. Köhler, D. Staehle, and P. Tran-Gia, *Fast heuristics for optimal routing in large IP networks*, Tech. report, Department of Computer Science, University of Würzburg, 2000.

[DFN] *DFN-Verein zur Förderung eines Deutschen Forschungsnetzes*, Information available at http://www.dfn.de.

[ERP01] M. Ericsson, M.G.C. Resende, and P.M. Pardalos, *A genetic algorithm for the weight setting problem in OSPF routing*, Tech. report, AT&T Labs Research, 2001.

[FGL+00] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, *NetScope: Traffic engineering for IP networks*, IEEE Network **14** (2000), no. 2, 11 – 19.

[FT00] B. Fortz and M. Thorup, *Internet traffic engineering by optimizing OSPF weights*, Proceedings of IEEE INFOCOM 2000, 2000.

[HK02] C. Helmberg and K.C. Kiwiel, *A spectral bundle method with bounds*, Mathematical Programming **93** (2002), no. 2, 173–194.

[HR00] C. Helmberg and F. Rendl, *A spectral bundle method for semidefinite programming*, SIAM Journal on Optimization **10** (2000), no. 3, 673–696.

[HY01] K. Holmberg and D. Yuan, *Optimization of internet protocol network design and routing*, Tech. report, Linköping University, 2001.

[ILO02] ILOG CPLEX Division, 889 Alder Avenue, Suite 200, Incline Village, NV 89451, USA, *ILOG CPLEX 8.0 reference manual*, 2002, Information available at http://www.cplex.com.

[LW93] F.Y.S. Lin and J.L. Wang, *Minimax open shortest path first routing algorithms in networks suporting the SMDS service*, Tech. report, Bell Communications Research, 1993.

[Pry02] M. Prytz, *On optimization in design of telecommunications networks with multicast and unicast traffic*, Ph.D. thesis, Royal Institute of Technology, Stockholm, Sweden, 2002.

[Wes00] R. Wessäly, *Dimensioning survivable capacitated networks*, Ph.D. thesis, Technical University Berlin, Germany, 2000.