Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

Sebastian Orlowski
Arie M.C.A. Koster
Christian Raack
Roland Wessäly

Two-layer Network Design by Branch-and-Cut featuring MIP-based Heuristics

ZIB-Report 06-47 (November 2006)

Abstract

This paper deals with MIP-based primal heuristics to be used within a branch-and-cut approach for solving multi-layer telecommunication network design problems. Based on a mixed-integer programming formulation for two network layers, we present three heuristics for solving important subproblems, two of which solve a sub-MIP. On multi-layer planning instances with many parallel logical links, we show the effectiveness of our heuristics in finding good solutions early in the branch-and-cut search tree.

Keywords: multi-layer network design, integer programming, branch-and-cut, heuristics
MSC classification (2000): 90C57, 90C59, 90B18

1 Introduction

Many telecommunication networks consist of several technological layers, like MPLS, ATM, SDH, or WDM, which are strongly interdependent. In an SDH over WDM network, for instance, lightpaths with different bandwidths (e. g., 2.5, 10, or 40 Gbit/s) are configured between the nodes of the network. These lightpaths are routed through a fiber network, where each fiber supports up to 40 or 80 lightpaths using wavelength division multiplexing. In a leased-line SDH network, STM-1 or STM-4 connections are configured between the nodes with a bandwidth of 155 and 622 Mbit/s, respectively. These connections may be realized using fibers or capacitated radio links. For instance, if the end-nodes of such a link are equipped with a 4xSTM-1 or 1xSTM-4 line card, the link can be used by four STM-1 connections or by one STM-4 connection, respectively.

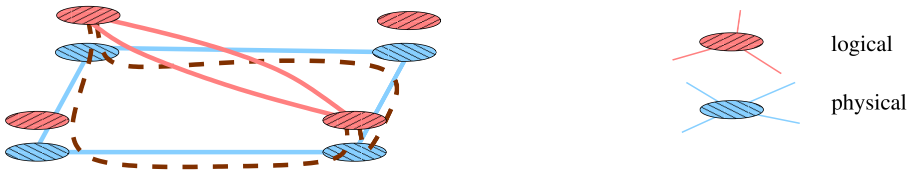

Figure 1: Links (solid) in upper logical layer correspond to paths (dashed) in the physical layer.

A two-layer network can be modeled using a lower-layer *physical* graph representing for example a fiber or radio network, and an upper-layer *logical* graph with the same set of nodes (or a subset of it), whose links represent the lightpath or STM-N connections (see Figure 1). Every link in the logical graph corresponds to a path in the physical network. Capacity modules, i. e., different bandwidths, installed on a logical link consume fiber or port capacity on the corresponding physical links. The planning task is to install discrete capacities on logical and physical links at minimum cost such that communication demands can be routed through the logical links and the physical network supports the logical link configuration. The process of choosing a subset of all possible logical links together with their physical representation is called *grooming*.

Several authors have used mixed-integer programming (MIP) models for two-layer network design. Some of them have employed a branch-and-cut solver as a black box either to solve their problems directly [3] or to obtain dual bounds to assess the quality of their iterative heuristic approaches [7, 9]. Others have added cutting planes or additional variables to Cplex on the fly [5, 6]. Primal heuristics, however, have always been used separately from branch-and-cut in multi-layer network design, although modern MIP solvers also provide callbacks for integrating problem-specific heuristics into the branch-and-cut process.

In this paper, we propose to combine primal heuristics and branch-and-cut instead of viewing these approaches as competing alternatives. We repeatedly solve two important subproblems as heuristics within a branch-and-cut procedure to obtain both high-quality solutions and a dual bound at the same time. One of these subtasks is the *grooming and capacity assignment* problem for a given flow, i. e., choosing a subset out of a possibly large set of admissible logical and physical links and installing capacities on them. The other one is the *routing* subproblem, i. e., determining a routing within a given capacity configuration. These subtasks are themselves formulated as mixed-integer programs and solved using a branch-and-cut algorithm, allowing for an easy extension of the model by further side constraints without major implementation changes.

This paper is structured as follows. In the next section, we present a mixed-integer programming formulation and review the corresponding literature. The following section describes our algorithmic approach and our MIP-based primal heuristics. On multi-layer planning instances with many parallel logical links derived from six SNDlib [1] networks, we show their effectiveness in finding good solutions early in the branch-and-cut search tree, and investigate the effect of other MIP-based heuristics implemented in the branch-and-cut solver SCIP [2]. We conclude by indicating some further research directions.

2 Mixed-integer programming model

For solving the two-layer network planning problem, we use the following mixed-integer programming formulation. The physical network is represented by an undirected graph (V, E). The logical network is modeled by an undirected graph (V, L) with the same set of nodes and a fixed set L of admissible logical links, where each $\ell \in L$ is defined by an undirected path in the physical network. Consequently, there can be many parallel logical links corresponding to different physical paths between the same nodes, and the size of L can be exponential in $|V|$. For ease of notation, let $L_e \subseteq L$ be the set of logical links containing physical link $e \in E$. Similarly, $L_{vw} \subseteq L$ denotes the subset of all logical links connecting nodes v and w. Furthermore, a set Q of commodities is given, where each commodity $q \in Q$ has a demand value d_v^q at node $v \in V$ such that $\sum_{v \in V} d_v^q = 0$. Typically, these commodities are derived by aggregating point-to-point demands at a common source or target node. Each logical link $\ell \in L$ has a set M_ℓ of installable capacity modules (e. g., lightpath capacities) which can be installed in arbitrary combination. Each module $m \in M_\ell$ has a capacity of C^m and a cost of c^m. On a physical link $e \in E$, any number of capacity batches (e. g., fibers) can be installed at a unit cost of c_e, each of them supporting at most B logical capacity modules.

For a logical link $\ell \in L$ and a module $m \in M_\ell$, the logical link capacity variable $y_\ell^m \in \mathbb{Z}_+$ represents the number of modules of type m installed on ℓ. Similarly, the number of batches installed on physical link $e \in E$ is given by the physical link capacity variable $x_e \in \mathbb{Z}_+$. Finally, the flow variables $f_{\ell,vw}^q, f_{\ell,wv}^q \in \mathbb{R}_+$ represent the flow for commodity $q \in Q$ on logical link $\ell \in L_{vw}$ directed from v to w and vice versa. The goal is to minimize total installation cost (1), subject to the constraints that all demands can be routed in the logical network (2) and that neither logical (3) nor physical link capacities (4) are exceeded:

$$\min \quad \sum_{\ell \in L} \sum_{m \in M_\ell} c^m y_\ell^m + \sum_{e \in E} c_e x_e \tag{1}$$

$$\text{s.t.} \quad \sum_{w \in V} \sum_{\ell \in L_{vw}} (f_{\ell,vw}^q - f_{\ell,wv}^q) = d_v^q \qquad v \in V,\ q \in Q \tag{2}$$

$$\sum_{m \in M_\ell} C^m y_\ell^m - \sum_{q \in Q} (f_{\ell,vw}^q + f_{\ell,wv}^q) \geq 0 \qquad v, w \in V,\ \ell \in L_{vw} \tag{3}$$

$$B x_e - \sum_{\ell \in L_e} \sum_{m \in M_\ell} y_\ell^m \geq 0 \qquad e \in E \tag{4}$$

$$y_\ell^m, x_e \in \mathbb{Z}_+, \quad f_{\ell,vw}^q, f_{\ell,wv}^q \in \mathbb{R}_+ \tag{5}$$

Similar mixed-integer programming models for multi-layer network design problems have been proposed by several authors. Some of them design the logical and physical network at the same time [7, 9], while others assume the physical network to be given [3, 5, 6].

Kubilinskas and Pióro [9] present an integer programming formulation where one capacity type can be installed in integer multiples on the logical and physical links, respectively. Their iterative search procedure repeatedly reroutes the demands and solves a sub-MIP to install as much logical and physical capacity as needed, taking global budget constraints into account. The solutions obtained by this heuristic are compared to branch-and-cut results on one network instance. In the model of H"oller and Vo"s [7], SDH or WDM capacities can be installed on the logical links in multiples of a wavelength granularity. Their formulation comprises an unsplittable routing of wavelength demands on the logical links as well as the dimensioning of a suitable fiber network. The proposed GRASP-like heuristic iteratively sorts the demands in different ways, reroutes bundles of demands, and recomputes the necessary capacities. On small instances, dual bounds are obtained from a separate branch-and-cut run.

With a fixed physical layer, the planning task is to determine a demand routing and a suitable logical network supported by the physical link capacities. For a model with one base capacity on the logical links, Dahl et al. [5] propose a branch-and-cut algorithm with different cutting planes, such as knapsack, strengthened cutset, flow-cutset, and hypomatchable inequalities. Baier et al. [3] present a mixed-integer programming model for multi-period planning with node-switching capacities, which is solved using Cplex as a black box. In contrast to most other multi-layer models, their formulation allows for several logical link capacities to support lightpaths with bandwidth 2.5, 10, and 40 Gbit/s in the same network. The path-flow formulation presented in Dawande et al. [6] includes upper bounds on the number of logical links (wavelength channels) at each node to account for a limited number of transponders. It is solved using a price-and-branch algorithm where routing paths are generated in the root relaxation. A wavelength assignment is determined heuristically afterwards. Dual bounds are obtained from a Lagrangian relaxation of an equivalent edge-flow formulation.

3 Branch-and-cut algorithm with MIP-based heuristics

We solve the mixed-integer programming formulation using the branch-and-cut framework SCIP [2]. In addition, we have implemented several heuristics to construct feasible network configurations based on integer or fractional solutions. At every node of the search tree, SCIP generates cutting planes and calls both our heuristics and some of its own ones to identify feasible integer solutions. If a new best solution is identified, it is added to SCIP's solution pool such that it can be used by other heuristics which take feasible solutions as a basis for their work. We will now describe our heuristics and their use within the branch-and-cut framework.

Our MIP-based heuristics address two major subtasks. GroomCapMip and GroomCapHeur solve the grooming and capacity installation subproblem for a given routing exactly and heuristically, respectively, whereas ReroutingMip computes a routing within certain link capacities, trying to reduce the required capacity at the same time. By construction, the MIP-based heuristics can easily be adapted to include additional planning requirements, such as node hardware or survivability constraints.

GroomCapMip The GroomCapMip procedure addresses the grooming and capacity assignment subproblem for a given routing by solving a MIP. Let $f_\ell^* := \sum_{q\in Q}(f_{\ell,vw}^q + f_{\ell,wv}^q)$ be the total flow on logical link $\ell \in L_{vw}$ in an integer or LP solution (after removing possible cycle flows). We construct a sub-MIP of the original formulation (1)–(5) that contains logical and physical capacity variables but no routing information:

$$\min \Big\{ (1) \mid (4), \sum_{m\in M_\ell} C^m y_\ell^m \geq \lceil f_\ell^* \rceil \quad \forall \ell \in L, \;\; x_e, y_\ell^m \in \mathbb{Z}_+ \Big\}.$$

Using SCIP's branch-and-cut algorithm, this sub-MIP is solved as an improvement heuristic every time a new best solution is identified, trying to reduce link capacity cost based on the given routing. As the focus of the sub-MIP is on feasible solutions and not on the lower bound, we disable cut generation and expensive heuristics in the subproblem and impose limits of 30 seconds and 10000 branch-and-bound nodes.

GroomCapHeur In contrast to the GroomCapMip algorithm that solves the grooming and capacity assignment problem exactly, the fast and simple GroomCapHeur procedure addresses this problem heuristically by decomposition. Again, let f_ℓ^* be the total flow on logical link $\ell \in L$ in an integer or LP solution after removing cycle flows. Installing capacities on ℓ at minimum cost with a lower bound of f_ℓ^* can be formulated as an integer knapsack problem:

$$\min \Big\{ \sum_{m \in M_\ell} c^m y_\ell^m \mid \sum_{m \in M_\ell} C^m y_\ell^m \geq \lceil f_\ell^* \rceil \,, \;\; y_\ell^m \in \mathbb{Z}_+ \Big\}.$$

For $|M_\ell| = 1$ this knapsack problem is trivial to solve. Otherwise, it is solved heuristically for each logical link $\ell \in L$ using a greedy algorithm. In a second step, each physical link is equipped with the necessary number of capacity batches to accommodate the computed logical link capacities y^* by setting $x_e := \left\lceil \frac{1}{B} \sum_{\ell \in L_e} \sum_{m \in M_\ell} y_\ell^{m*} \right\rceil$. As this heuristic runs very fast, we call it at every branch-and-cut node to construct feasible solutions from the current LP solution.

ReroutingMip The ReroutingMip heuristic determines a routing together with a minimum-cost capacity installation subject to an upper capacity bound on the logical links. More precisely, given an upper bound U_ℓ^* on the capacity of each logical link $\ell \in L$, ReroutingMip solves the following problem using SCIP's branch-and-cut capabilities:

$$\min \Big\{ (1) \mid (2)\text{–}(5), \;\; \sum_{m \in M_\ell} C^m y_\ell^m \leq U_\ell^* \;\; \forall \ell \in L \Big\}.$$

With small U_ℓ^*, this problem is much easier to solve than the original problem. By setting U_ℓ^* to the total capacity of link $\ell \in L$ in an integer solution, ReroutingMip can be used as an improvement algorithm that tries to reduce capacities by rerouting flow. This generalizes the rerouting step in the iterative heuristics proposed in [7, 9], making it independent of the ordering of the demands.

We employ ReroutingMip not as an improvement heuristic but as a construction algorithm. Given some value $\kappa \geq 1$ and an LP solution with total logical link capacities $y_\ell^* := \sum_{m \in M_\ell} C^m y_\ell^{m*}$, we solve the above sub-MIP with $U_\ell^* := C^0 \left\lceil \frac{\kappa}{C^0} y_\ell{}^* \right\rceil$ where C^0 is the smallest module capacity installable on ℓ. If the installable capacities form a divisibility chain (which is often the case in practical applications), U_ℓ^* is the smallest installable integer capacity greater than or equal to κy_ℓ^*. Obviously, a higher value of κ augments the solution space of the subproblem, allowing for better solutions but also making it harder to solve. Experimenting with different values, we found that $\kappa = 2$ often allowed to quickly determine good solutions in the sub-MIP.

As the ReroutingMip algorithm consumes much more time than the other heuristics, we restrict its application to the LP solution at the end of the branch-and-cut root node. In the sub-MIP (as well as in the original problem), good solutions are often found within the first few branch-and-bound nodes, whereas much time is spent afterwards on proving optimality of the solution. Hence, we disable cut generation and expensive heuristics in the subproblem, and we impose a time limit of three minutes, a node limit of 2000 nodes, and a gap limit of 2 %. To increase the chance of finding good solutions, we also apply the GroomCapHeur and GroomCapMip algorithms within the sub-MIP, which tends to improve the overall solution quality.

4 Preliminary results

We have integrated our algorithms as heuristic callbacks into the branch-and-cut framework SCIP 0.90 [2], using Cplex 10.1 [8] as the underlying LP solver. All computations were performed on a Linux machine with a Pentium IV 3.8 GHz processor and 2 GB of memory, using a time limit of one hour.

Table 1 summarizes our 48 multi-layer planning instances based on six SNDlib [1] networks which have been taken as the physical graph. Using the published communication demands, we have constructed eight test instances from each of these networks by computing different sets of

instance	nodes	physical links	demands	logical links	$\lvert M_\ell \rvert$	B
Polska	12	18	66	66–2441	2	4
Nobel-us	14	21	91	91–4526	1	40
Nobel-germany	17	26	121	136–6533	1	40
France	25	45	300	300–5892	1	40
Nobel-eu	28	41	378	378–7560	1	40
Pioro40	40	89	780	780–7800	2	4

Table 1: Characteristics of the test instances

admissible logical links. For every $k \in \{1, 2, 5, 10, 20, 30, 40, 50\}$, a set L has been determined by letting k_{vw} be the largest number of existing v-w-paths less than or equal to k and computing k_{vw} shortest physical paths (w. r. t. km-length) between every pair of nodes v, w. This approximates the fact that in principle, every possible physical path might define a logical link (subject to technological limitations), and significantly augments the solution space compared to numerous publications that assume at most one logical link between each pair of nodes [3, 5–7, 9]. For the France, Polska, and Pioro40 instances, we have used the capacity and cost structures from SNDlib. For the Nobel instances, we have assumed the WDM-based cost model from [10] developed together with the German network provider T-Systems, which takes cost of fibers, ports, and regenerators into account.

We ran four tests on each instance: one with the default settings of SCIP, one with GroomCapMip and GroomCapHeur, one with ReroutingMip, and one with all three heuristics. The following results are based on the 150 runs (38, 37, 38, and 37 for the four settings, respectively) where at least the root node was finished within the time limit, including cutting plane generation and root heuristics. Table 1 shows the range of the number of logical links of those test instances.

It turned out that the number of admissible logical links had little influence on the optimality gap after one hour. The final gap was below 5 % in most of the Pioro40, Polska, Nobel-germany, and Nobel-us instances even with several thousands of logical links. The Nobel-eu and France instances, which have relatively large logical link capacities compared to the demand values, were more difficult to solve with gaps around 20 % and 100 %, respectively. In some longer test runs on these instances, we found that their gaps could still significantly be reduced just by raising the lower bound. Averaged over all instances, our heuristics had little impact on the final gap, but helped to find high-quality solutions earlier in the search tree.

The ReroutingMip algorithm was particularly successful in terms of solution quality. In 20 out of the 75 runs where it was called, this heuristic identified a better solution at the root node than all other heuristics (both ours and SCIP's) within one hour. In four cases, this was the only solution found at all. In fact, we observed that most logical links used in good integer solutions also had a nonzero capacity in the LP solution at the end of the root node, and many logical links not used in feasible solutions were also unused in the root LP solution. In contrast, the LP routings were often completely different from those in feasible solutions. It was also these observations which motivated the approach of rerouting flow based on the root LP capacities.

The GroomCapMip heuristic could improve 10 other solutions during 75 runs. In seven of these cases, the improvement took place at the root node, and the resulting solution was the best one found within one hour. In most cases, the GroomCapMip algorithm took less than one second of computation time. However, in some of the Polska and Pioro40 instances (where $|M_\ell| = 2$), the sub-MIP was not solved within its time limit of 30 seconds. Consequently, calling it as a construction heuristic at every node would be too time-consuming, but as an improvement heuristic for feasible integer solutions, its benefit outweighs the effort.

Whereas the previous heuristics invest much time in the solution quality, GroomCapHeur has been designed to construct reasonable solutions within short time. Consequently, this heuristic identified a feasible solution in the root node of all 75 runs where it was called, and it often detected several further incumbent solutions during the first branch-and-cut nodes. However, SCIP's own heuristics usually identified better solutions later on. As a fast heuristic, GroomCapHeur is thus particularly suited for large instances where only a very small number of branch-and-cut nodes is solved. In instances where many branch-and-cut nodes were solved, this heuristic often consumed about five minutes in total. Reducing the call frequency or imposing a maximum call depth in the search tree might be appropriate in such cases.

Summarizing, all of our heuristics were helpful in finding solutions early in the search tree. In particular, ReroutingMip and GroomCapMip often detected solutions at the root node that could not be improved by any other heuristic within one hour. With all our heuristics enabled, the best known solution was found at the root node in 15 out of the 37 runs, compared to only 8 cases with the default settings.

It may be interesting to note that also two sub-MIP-based heuristics of SCIP were quite successful. One of them is the Rens heuristic [4] which solves a sub-MIP for finding an optimal rounding of the root relaxation. In the default settings, it determined the best known solution in 8 out of the 38 runs. The other one is the Crossover heuristic [4] that fixes all variables having the same value in two feasible solutions, and solves the remaining problem as a sub-MIP. It detected 7 best solutions with all of our heuristics, compared to 4 with the default settings. Apparently, our heuristics were also useful as an input for Crossover.

5 Conclusions

Based on a mixed-integer programming model for two-layer network design, we have presented three primal heuristics to be called within a branch-and-cut algorithm. Two of them solve a restriction of the original formulation as a sub-MIP. On multi-layer planning instances with many parallel logical links, our heuristics significantly help the branch-and-cut solver in finding high-quality solutions early in the search tree.

For future research, it might be interesting to employ an iterative procedure like those in [7,9] as a primal heuristic at the root LP relaxation of a branch-and-cut algorithm. The subproblems can be solved either using a sub-MIP with time and node limits or with any combinatorial heuristic.

Acknowledgment

This research has been supported by the DFG Research Center Matheon and by the European COST Action 293 "Graphs and Algorithms in Communication Networks."

References

[1] SNDlib 1.0–Survivable network design data library. `http://sndlib.zib.de`, 2005.

[2] T. Achterberg. SCIP 0.90–Solving Constraint Integer Programs. `http://scip.zib.de`, 2006.

[3] G. Baier, T. Engel, A. Autenrieth, and P. Leisching. Mehrperiodenplanung optischer Transportnetze. In *7. ITG-Fachtagung Photonische Netze, Leipzig, Germany*, volume 193, pages 153–160. VDE-Verlag, April 2006.

[4] T. Berthold. Primal heuristics for mixed integer programs. Diploma thesis, Technische Universität Berlin, September 2006.

[5] G. Dahl, A. Martin, and M. Stoer. Routing through virtual paths in layered telecommunication networks. *Operations Research*, 47(5):693–702, 1999.

[6] M. Dawande, R. Gupta, S. Naranpanawe, and C. Sriskandarajah. A traffic-grooming algorithm for wavelength-routed optical networks. *INFORMS Journal on Computing*, 2006. To appear.

[7] H. Höller and S. Voß. A heuristic approach for combined equipment-planning and routing in multi-layer SDH/WDM networks. *European Journal of Operational Research*, 171(3):787–796, June 2006.

[8] ILOG CPLEX Division, 889 Alder Avenue, Suite 200, Incline Village, NV 89451, USA. *CPLEX 10.1 Reference Manual*, 2006. `http://www.cplex.com`.

[9] E. Kubilinskas and M. Pióro. An IP/MPLS over WDM network design problem. In *Proceedings of the Second International Network Optimization Conference (INOC 2005), Lisbon*, volume 3, pages 718–725, March 2005.

[10] A. Zymolka. *Design of Survivable Optical Networks by Mathematical Optimization.* PhD thesis, Technische Universität Berlin, 2006. Submitted.