A Node-Based Layered Graph Approach for the Steiner Tree Problem with Revenues, Budget and Hop-Constraints

Markus Sinnl*[1] and Ivana Ljubić†[1]

[1]Department of Statistics and Operations Research, Faculty of Business, Economics and Statistics, University of Vienna, Austria

Abstract

The Steiner tree problem with revenues, budget and hop-constraints (STPRBH) is a variant of the classical Steiner tree problem. The goal is to find a tree maximizing the collected revenue, which is associated with nodes, subject to a given budget for the edge cost of the tree and a hop-limit for the distance between the given root node and any other node in that tree.

In this work, we introduce a novel generic way to model hop-constrained tree problems as integer linear programs and apply it to the STPRBH. Our approach is based on the concept of layered graphs that gained widespread attention in the recent years, due to their computational advantage when compared to previous formulations for modeling hop-constraints. Contrary to previous MIP formulations based on layered graphs (that are arc-based models), our model is node-based. Thus it contains much less variables and allows to tackle large-scale instances and/or instances with large hop-limits, for which the size of arc-based layered graph models may become prohibitive. The aim of our model is to provide a good compromise between quality of root relaxation bounds and the size of the underlying MIP formulation.

We implemented a branch-and-cut algorithm for the STPRBH based on our new model. Most of the instances available for the DIMACS challenge, including 78 (out of 86) previously unsolved ones, can be solved to proven optimality within a time limit of 1000 seconds, most of them being solved within a few seconds only. These instances contain up to 500 nodes and 12500 edges, with hop-limit up to 25.

1 Introduction

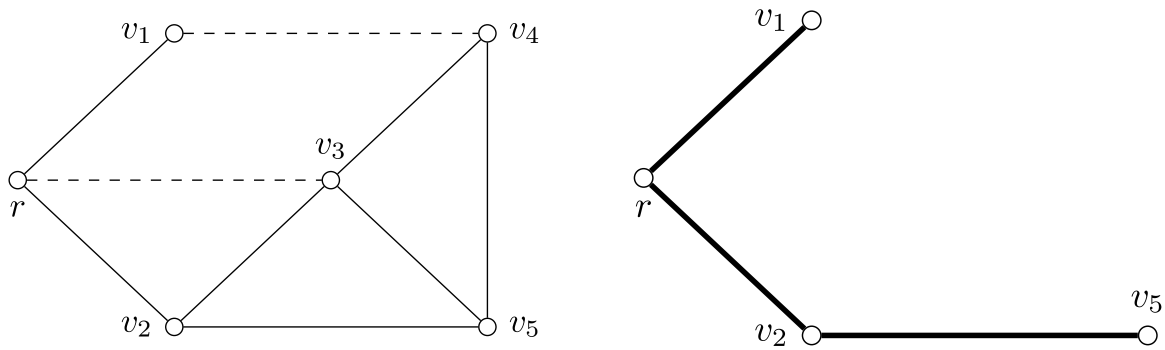
The Steiner tree problem in graphs (SPG) is a classical problem in operations research, see e.g., [20, 21, 29] and the references therein. In the SPG, we are given a graph $G(V,E)$ with edge costs $c : E \mapsto \mathbb{R}^+$ and a set of terminals $T \subseteq V$, and the goal is to find a tree of minimal cost, which contains all terminals. In this work, we consider a variant of the SPG known as the Steiner tree problem with revenues, budget and hop-constraints (STPRBH), whose definition is given below. The problems has been intensively studied in the last years using exact [7, 22, 27] and heuristic [6, 13, 14] approaches.

Definition 1 (The Steiner Tree Problem with Revenues, Budget and Hop-Constraints (STPRBH)). *We are given an undirected graph $G = (V,E)$ with edge costs $c : E \mapsto \mathbb{R}^+$, node revenues $p : V \mapsto \mathbb{R}^+$, a dedicated root node $r \in V$, a hop-limit $H \in \mathbb{N}^+$ and a budget limit $B \in \mathbb{R}^+$.*

A feasible solution of the STPRBH is a subtree $\mathcal{T} = (V_S \subseteq V, E_S \subseteq E)$ rooted at r, where every node in V_S can be reached form the root r using at most H edges and the total cost of the edges in E_S does not exceed B, i.e., $\sum_{e \in E_S} c_e \leq B$. The goal is to find a feasible subtree $\mathcal{T}^*$ that maximizes the revenue defined as $\sum_{v \in V_S} p_v$.

Figure 1 depicts an instance of the STPRBH and its optimal solution.

*markus.sinnl@univie.ac.at
†ivana.ljubic@univie.ac.at



(a) Instance.

(b) Solution.

Figure 1: (1a) Graph of an instance of the STPRBH problem. Let $p_1 = 10, p_2 = 0, p_3 = 4, p_4 = 9, p_5 = 5$, the cost of the solid edges be one, and of the dashed edges be five. (1b) The optimal solution for $H = 2$ and $B = 3$ has objective value 15.

Our Contribution. In this work, we present a novel generic way to model hop-constrained tree problems as integer linear programs (ILPs) and apply it to the STPRBH. Our approach is based on layered graphs, a concept which has gained widespread attention in the last few years. On the one hand, layered graphs allow for significant improvements of computing times when compared to previously available extended formulations (see [19]). On the other hand, they are also shown to theoretically dominate most of the available extended formulations that model hop-constraints. Instead of modeling the problem on G, a layered graph is constructed such that for each layer $1 \leq h \leq H$, a copy of the nodes of G is established, and nodes between two consecutive layers are connected whenever there exists a connection between them in G (for more details, see Section 2). The underlying problem is then formulated as a Steiner arborescence problem using arc variables on such obtained layered digraph. While this formulation often provides very good LP-bounds (see, e.g. [19]), the number of variables (which is $O(H|E|)$), often becomes prohibitive when the problem is formulated on larger graphs, or when larger hop-limits H are considered.

To overcome this latter drawback, we apply the "thinning out" idea recently exploited in [10, 11] for solving Steiner trees and facility location problems, respectively. In our approach, the set of arc variables of the layered graph is projected out, resulting in two new formulations that comprise only node variables on the layered graph (along with node and arc variables on G). Whereas the standard layered graph approach involves $O(H|E|)$ variables, our new models deal with $O(H|V| + |E|)$ variables only. One of our models is compact, i.e., it requires only a polynomial number of constraints to ensure connectivity of the solution. However, we show that better bounds can be obtained by imposing an exponential number of subtour elimination constraints on G. In both cases, our models provide a good compromise between quality of obtained LP-bounds and the size of the underlying model.

A branch-and-cut-algorithm for the STPRBH derived from our new formulation solves most of the instances from the DIMACS Challenge [8] to provable optimality in a short time (often within a few seconds). This includes 78 (out of 86) instances for which the optimal solution has been previously unknown. Our framework won the category STPRBH in the challenge. The program is made available online under `http://homepage.univie.ac.at/markus.sinnl/program-codes/stprbh/`.

Outline of the Paper. Our paper is structured as follows: In Section 2, a short review of layered graphs is followed by the presentation of our generic new model together with valid inequalities. Section 3 contains a description of our solution framework, including a preprocessing phase and primal heuristics. Computational results are presented in Section 4. Section 5 concludes the work with a short summary and a discussion of future work. It points out a broader potential of the proposed "thinning out" approach for modeling hop- or diameter-constrained trees.

Previous Work. The STPRBH has been introduced by [7] where three branch-and-cut approaches have been presented: one based on Miller-Tucker-Zemlin constraints, one on Dantzig-Fulkerson-Johnson (also known as subtour-elimination) constraints, and one on hop-indexed formulation. Note that the latter formulation is based on hop-indexed edge variables, i.e., it can be viewed as a compact arc-based MIP formulation on a layered graph. Instances derived from sets B and C of the OR-library [2] have also been introduced in [7]. All instances from the set B and instances $C1$ to $C5$ have been solved to optimality with the approaches from [7]. These instances contain 500 nodes and 625 edges. However, the authors of [7] have demonstrated that no single model works well for all instances. In [6], the same authors proposed a greedy heuristic and a tabu search with some improvement procedures. They also reported some results for $C6$ to $C20$. These instances consist of 500 nodes and up to 12500 edges. According to [6], for these instances, not even the root relaxation of the models presented in [7] could be solved within a time limit of two hours (in most of the cases). Branch-and-price approaches for the STPRBH have been studied in Master Thesis of Sinnl [27]. A lifted Miller-Tucker-Zemlin formulation and a formulation based on reformulation-linearization techniques were given in [22]. The two latter works provide computational results on the instances from sets B and $C1$ to $C5$, but offer no consistent speed up, when compared to [6]. Recently, a breakout local search algorithm (see [13]) and a memetic algorithm (see [14]) have been proposed. These two recent papers provide improved feasible solutions for some of the unsolved instances ($C6$ to $C20$). Some new instances based on graphs $C16$ to $C20$ are also introduced in [14].

2 Problem Formulation and Valid Inequalities

Let $G_L = (V_L, A_L)$ be the layered graph associated with a rooted graph $G(V,E)$ and hop-limit H. It is defined as follows (see, e.g., [19]): The node set $V_L = r \cup V^1 \cup V^2 \cup \ldots \cup V^H$, where V^h contains a copy v^h of all nodes $v \in V \setminus \{r\}$. Note that the root node r is the only node at layer zero. The arc set $A_L = A^1 \cup A^2 \cup \ldots \cup A^H$, where A^h contains a directed copy (i,j) of an edge $\{i,j\} \in E$, iff $i \in V^{h-1}$ and $j \in V^h$.[1] Thus the layered graph has size $O(H(|V|+|E|))$. Figure 2 shows the layered graph associated with our exemplary instance from Figure 1a and $H = 3$.

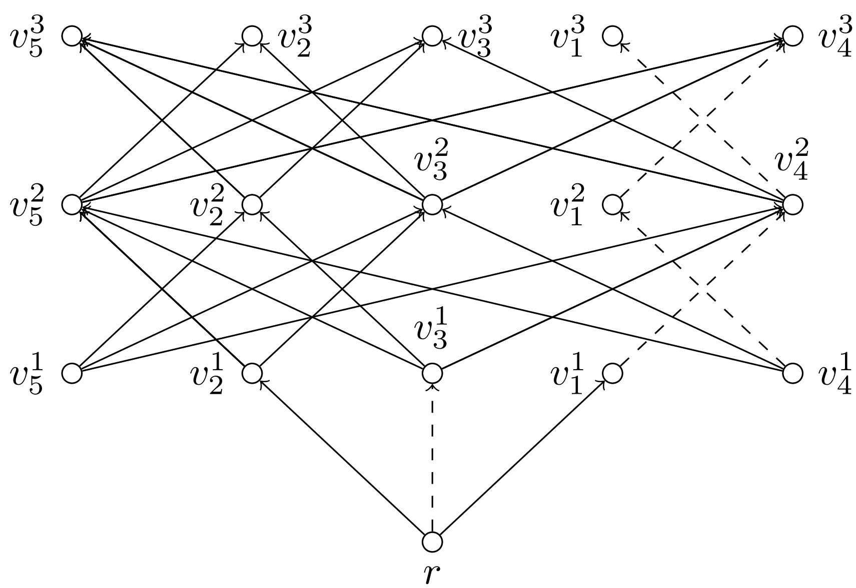


Figure 2: Layered graph associated with the graph from (1a) and $H = 3$.

It has been shown in [19] that the optimal hop-constrained spanning/Steiner tree problem can be obtained by solving the Steiner tree problem on the layered graph G_L with additional constraints that each Steiner/terminal node v has to be visited at most/exactly once across all layers. To this end, hop-constrained problems are formulated on

[1] Observe that in the definition in [19], there is an additional set of arcs going from a node v^h on any layer $1 \leq h \leq H-1$ to its corresponding node v^H on the last layer, we do not need these arcs in our approach.

G_L by associating variables to the arcs A_L of the layered graph, e.g., x_{ij}^h is one, if arc $\{i, j\}$ is used on layer h (see, e.g., [19, 23]). While this usually gives models with strong LP-bounds, the size of the resulting MIP formulations soon becomes prohibitive. We thus propose to project out arc variables from the layered graph and model the hop-constraints by associating variables with the nodes V_L of the layered graph.

To do so, we transform the graph G into a rooted digraph $D = (V, A)$, where A are the bidirected edges from E (incoming arcs to the root node are removed). We use the following sets of binary variables to model our problem (resp., generic hop-constraint trees)

$$x_a = \begin{cases} 1 & \text{if arc } a \text{ is part of the solution} \\ 0 & \text{otherwise} \end{cases} \quad \text{for } a \in A;$$

$$y_v = \begin{cases} 1 & \text{if node } v \text{ is part of the solution} \\ 0 & \text{otherwise} \end{cases} \quad \text{for } v \in V;$$

$$y_v^h = \begin{cases} 1 & \text{if node } v \text{ is on layer } h \text{ in the solution} \\ 0 & \text{otherwise} \end{cases} \quad \text{for } v \in V \setminus \{r\}, 1 \leq h \leq H.$$

For $1 \leq i \leq H-1$, let $H_i = \{i, \ldots, H\}$. Furthermore, let $\delta^-(W) = \{(i, j) \in A : i \notin W, j \in W\}$ and $\delta^+(W) = \{(i, j) \in A : i \in W, j \notin W\}$. Let $T = \{v \in V : p_v > 0\}$ and $S = V \setminus \{T \cup \{r\}\}$. It can be easily seen, that there always exists an optimal solution to the STPRBH, where only nodes from T are leaf nodes. We will refer to T as set of terminals, and S as Steiner nodes.

Using this notation, we obtain a generic set of inequalities for modeling hop-constrained tree problems, denoted by (NODEHOP):

(NODEHOP)

$$x(\delta^-(W)) \geq y_v \qquad \forall W \subseteq V, v \in W \cap T, r \notin W \qquad \text{(CCuts)}$$

$$y_r = 1 \qquad \text{(Root)}$$

$$x(\delta^-(v)) = y_v \qquad \forall v \in V \setminus \{r\} \qquad \text{(Indegr)}$$

$$\sum_{h \in H_1} y_v^h = y_v \qquad \forall v \in V \setminus \{r\} \qquad \text{(NH-Link)}$$

$$x_{rv} = y_v^1 \qquad \forall (r, v) \in A \qquad \text{(Root-Link)}$$

$$y_v^{h-1} + x_{vw} \leq 1 + y_w^h \qquad \forall (v, w) \in A, v \neq r, h \in H_2 \qquad \text{(HLink-)}$$

$$y_v^H + x_{vw} \leq 1 \qquad \forall (v, w) \in A, v \neq r \qquad \text{(HEnd-)}$$

$$(x_a, y_v, y_v^h) \in \{0, 1\}^{|A| \times |V| \times H|V \setminus \{r\}|} \qquad \text{(Binary)}$$

Constraints (CCuts), (Root) and (Indegr) comprise the cut-set formulation for the (prize-collecting) Steiner tree problem (see, e.g. [24]) and ensure that our solution contains an arborescence rooted at r. The remaining set of inequalities (NH-Link)-(HEnd-) deals with the hop-constraint: *Node-hop link* inequalities (NH-Link) ensure that if a node is part of the solution, it must lie on some layer. *Hop-end* inequalities (HEnd-) make sure that if a node lies on layer H, there can be no outgoing arc from it. Moreover, if the arc going from the root to node v is used, node v must lie on layer 1, which is ensured by (Root-Link). *Hop-link* constraints (HLink-) make sure that if a node v lies on layer $h-1$ ($2 \leq h \leq H$) and arc (v, w) is taken in the solution, then node w must lie on layer h. Note that crucial for the validity of our model is the tree/arborescence property: since every node only has one incoming arc (see constraints (Indegr)), the layer of each node is uniquely defined. Thus, constraints (CCuts) to (Binary) ensure in a generic way that the solution is an arborescence, satisfying the hop-constraint.

Using the generic model NODEHOP, it is easy to obtain the following formulation for the STPRBH:

$$\text{(STPRBH)} \qquad \max \sum_{v \in T} p_v y_v \tag{obj}$$

$$\sum_{a \in A} c_a x_a \leq B \tag{Budget}$$

$$(x, y, y^h) \in NODEHOP$$

The objective function (obj) ensures maximization of the profit, while constraint (Budget) makes sure that a solution does not exceed the given budget B. Our model contains $|A| + (H+1)|V|$ variables, and an exponential number of connectivity constraints (CCuts). Next, we show that even a compact formulation obtained by replacing (CCuts) with a smaller family of constraints, provides a valid model for the STPRBH.

Theorem 1. *Let sNODEHOP denote the compact model obtained from NODEHOP by replacing constraints* (CCuts) *with generalized subtour elimination constraints of size two:*

$$x_{vw} + x_{wv} \leq y_w, \quad v, w \in V \tag{GSEC2}$$

This compact model is valid for the STPRBH.

Proof. Let $(x, y, y^h) \in$ sNODEHOP be the optimal solution of sNODEHOP and let *Sol* be the graph associated with this solution. We show that *Sol* is connected, does not contain cycles and does not violate the hop-limit.

In-degree constraints (Indegr), together with inclusion of the root (with in-degree zero), and constraints (GSEC2) ensure that the number of nodes in *Sol* is the number of arcs plus one. In-degree constraints ensure that there cannot be isolated nodes, except maybe the root node. *Sol* is cycle-free because each node has to be associated to exactly one layer h, $1 \leq h \leq H$. A cycle in *Sol* would imply (due to inequalities (HLink-)) that there will be a node v in the cycle with $y_v^h = y_v^\ell = 1$, for $1 \leq h < \ell \leq H$, which violates inequality (NH-Link). Hence, the resulting solution is cycle-free, with the number of edges being one less than the number of nodes, which implies that *Sol* is a tree.

It only remains to show that for every node $v \in Sol$, there exists a directed path from r to v using at most H arcs. Assume there is a node $v \in Sol$ for which the above condition does not hold. This is however a contradiction with constraints (HLink-) and (HEnd-), which concludes the proof. □

In the following we provide some valid inequalities for the proposed new model NODEHOP. Some of these inequalities are only valid, when $T \neq V$, i.e., for Steiner tree problems, but not for their spanning tree counterparts.

Hop-Link Inequalities First, note that in both constraints (HLink-) and (HEnd-), the value 1 can be down-lifted to y_v. The constraints still remain valid, since any of y_v^{h-1}, y_v^H and x_{vw} set to one also implies that y_v is set to one.

Moreover, for any arc (v, w), constraints (HEnd-) can be made redundant by replacing inequalities (HLink-) with a lifted version with y_v^H added to the left-hand-side. This is summarized in the following result:

Theorem 2. *Let $h \in H_2$ and $(v, w) \in A, v \neq r$. Then the* hop-link *inequality*

$$y_v^H + y_v^{h-1} + x_{vw} \leq y_v + y_w^h, \quad \forall (v, w) \in A, v \neq r \tag{HLink}$$

is valid for NODEHOP.

Moreover, if $H \geq 3$, for any arc (v, w), constraints (HLink-) *and* (HEnd-) *can be replaced by* (HLink) *for $2 \leq h \leq H$.*

Proof. First, we show the validity of the constraints, when they are added to NODEHOP (i.e., (HLink-) and (HEnd-) remain in the model). Observe that only one of $y_v^H + y_v^{h-1}$ can be one, due to (NH-Link). Suppose y_v^H is zero, then the inequality reduces to (HLink-). Suppose y_v^H is one, then x_{vw} must be zero due to (HEnd-).

Now we assume all (HLink-) and (HEnd-) are replaced by constraints (HLink). The first argument of the previous proof still works, for y_v^H zero, then the inequalities reduces to (HLink-). It now remains to show that the presence of the complete set of inequalities is enough to force x_{vw} to zero for the case that y_v^H is one. Thus, suppose both y_v^H and x_{vw} are one. Then for any constraint (HLink) for the given arc (v, w), the left-hand-side is two (and due to the assumption $H \geq 3$, there exist a least two constraints). However, due to (NH-Link), for only one h, we can have that y_w^h is one, and thus for only one constraint (HLink) the right-hand-side can be two, which is a contradiction. □ □

Generalized Hop-Link Inequalities Using constraint (NH-Link) corresponding to node v, inequality (HLink) for an arc (v,w), $v \neq r$ and a given layer $h : 2 \leq h \leq H$ can be rewritten as

$$x_{vw} \leq y_w^h + \sum_{k \in H_1, k \neq (h-1), k \neq H} y_v^k \tag{1}$$

It has the intuitive meaning that if arc (v,w) is in the solution, it either ends at layer h (and thus has started at layer $h-1$), or it must have started at some other layer smaller than H and other than $h-1$. Consider now another layer $l \neq h$: Inequality (1) is valid, because when arc (v,w) ends at layer l, it must have started at layer $l-1$ and there is y_v^{l-1} on the right-hand-side of (1).

To motivate the generalization of these inequalities, observe that when the arc (v,w) ends at some layer $\neq l,h$, the variable y_v^{l-1} must be zero in a valid solution. Moreover, when arc (v,w) ends at layer l, the variable y_w^l must be one in any feasible solution. Thus it follows that y_v^{l-1} can be replaced by y_w^l in constraint (1) and the constraint remains valid. Generalizing this idea further, we observe that for each layer $h \geq 2$, in the summation on the right-hand-side, we must either include y_v^{h-1} or y_w^h. This brings us to the following family of inequalities:

Theorem 3. *Let P be the family of binary functions $P = 2^{H_2}$, $p \in P$ and $(v,w) \in A, v \neq r$. Then the* generalized hop-link *inequality*

$$x_{vw} \leq \sum_{h \in H_2} \left(p_h y_v^{h-1} + (1-p_h) y_w^h \right) \tag{g-HLink}$$

is valid for NODEHOP.

Proof. Clearly, when node w lies on layer 1 it must be connected to the root node and x_{vw} must be zero. Thus suppose there exists a feasible solution, where node w lies on some layer $k : 2 \leq k \leq H$, x_{vw} is one, i.e., the arc (v,w) is used and the right-hand-side of (g-HLink) is zero. Since node w lies on layer k and the arc (v,w) is used, it follows that node v must lie on layer $k-1$. This implies that both y_v^{k-1} and y_w^k are one. Due to the definition of the function p, $p_k = 1$ or $p_k = 0$ and consequently, we have either y_v^{k-1} or y_w^k on the right-hand-side and thus the right-hand-side is one, which is a contradiction to the assumption that the inequality is violated. □ □

For each arc $(v,w) \in A$, constraints (g-HLink) can easily be separated in $O(H)$ time: Given a fractional solution $(\tilde{x}, \tilde{y}, \tilde{y}^h)$, for each layer $h \geq 2$, we consider the sum $\sum_{h \in H_2} \min\{\tilde{y}_v^{h-1}, \tilde{y}_w^h\}$. If the obtained sum is smaller than $\tilde{x}_{vw}$, a violated constraint is detected.

Let us now consider a pair of inequalities of type (g-HLink), one associated to (v,w) and the other to (w,v). Let $\hat{p}$ and $\tilde{p}$ be the functions from P defining the first and second inequality, respectively. Summing up this pair of inequalities, we obtain

$$x_{vw} + x_{wv} \leq \sum_{h \in H_3} \left((\hat{p}_h + (1 - \tilde{p}_{h-1})) y_v^{h-1} + (\tilde{p}_h + (1 - \hat{p}_{h-1})) y_w^{h-1} \right) + \hat{p}_2 y_v^1 + \tilde{p}_2 y_w^1 + (1 - \tilde{p}_H) y_v^H + (1 - \hat{p}_H) y_w^H$$

Thus, depending on the functions $\hat{p}$ and $\tilde{p}$, the coefficients of y_v^h and y_w^h are zero, one or two in the resulting inequality. Since $x_{vw} + x_{wv} \leq 1$ (this follows from inequalities (CCuts) and equalities (Indegr), respectively from inequalities (GSEC2)), all coefficients of value two can be down-lifted to one.

Thus, the validity of the new derived families of inequalities presented in the following two theorem follows immediately. The first set of inequalities is obtained with $\hat{p} = \tilde{p} = \begin{cases} 1 & \text{if } h \text{ is even} \\ 0 & \text{otherwise} \end{cases}$ and the second one with $\hat{p} = \tilde{p} = \begin{cases} 0 & \text{if } h \text{ is even} \\ 1 & \text{otherwise} \end{cases}$.

Theorem 4. *Let $(v,w) \in A$, $v \neq r$. Then the* odd two-arc hop-link *inequality*

$$x_{vw} + x_{wv} \leq \sum_{h \in H_1, h \text{ odd}} \left(y_v^h + y_w^h \right) \tag{o2AHLink}$$

is valid for NODEHOP.

Theorem 5. *Let $(v,w) \in A$, $v \neq r$. Then the* even two-arc hop-link *inequality*

$$x_{vw} + x_{wv} \leq \sum_{h \in H_2, h \text{ even}} \left(y_v^h + y_w^h\right) \tag{e2AHLink}$$

is valid for NODEHOP.

For each pair of arcs $(w,v),(v,w) \in A$, constraints (o2AHLink) and (e2AHLink) can easily be separated in $O(H)$ time in a similar fashion to inequalities (g-HLink).

Cut Inequalities on the Layered Graph If a node w lays on a layer h, there obviously must be at least one node $v \neq w$ at layer $h-1$ in the solution. This leads to the following family of *node-hop-index* inequalities:

$$\sum_{(v,w) \in A} y_v^{h-1} \geq y_w^h \tag{2}$$

Such inequalities (expressed in terms of arc-variables on the layered graph) are commonly used in the hop-indexed models for hop-constrained problems (see, e.g. [15]). They represent a compact way of ensuring a connectivity of a solution. However, these hop-indexed compact models are known to suffer from weak lower bounds. In state-of-the-art approaches, connectivity constraints are therefore modeled using cut-set inequalities on layered graphs (see, e.g. [19, 23]).

By considering a modified layered graph, where nodes are split into directed arcs, it is not hard to see, that one can consider cut-set inequalities derived on the layered graph using y^h and x variables. Preliminary computational experiments, however, showed that addition of such type of inequalities in general was not beneficial for our problem, due to the high cost of separation (which involves max-flow computations on this modified layered graph). Instead, we use a subfamily of these cut-set inequalities, *node-arc-cut-inequalities*, which we illustrate next.

Observe first that if the input graph is complete, node-hop-index inequalities will be in general very weak, since the left-hand-side contains all nodes on layer $(h-1)$ in this case. Clearly, also the following inequality holds for any $h \geq 2$ and node $w \neq r$, since it is a weaker version of inequalities (CCuts) for $W = \{w\}$:

$$\sum_{(v,w) \in A, v \neq r} x_{vw} \geq y_w^h. \tag{3}$$

Observe that in both (2) and (3), the right-hand-side is the same, and we sum over all arcs on the left-hand-side. Hence, we can derive a more general family of inequalities, which contains both (2) and (3) as a special case.

Theorem 6. *Let R be the family of functions $R = 2^A$ and $r \in R$, $w \in V$ and $2 \leq h \leq H$. Then the* node-arc-cut-inequalities

$$\sum_{(v,w) \in A, v \neq r} \left(r_{vw} x_{vw} + (1 - r_{vw}) y_v^{h-1}\right) \geq y_w^h \tag{NACut}$$

are valid for NODEHOP.

Proof. Suppose there exists a feasible solution, where y_w^h is one, i.e., node w lies on layer h, and the left-hand-side is zero. However, since the node lies on layer h, there must be an incoming arc (v,w) from some node v lying on layer $h-1$, thus both y_v^{h-1} and (v,w) must be one. One of these variables is on the left-hand-side of constraint (NACut), and thus the left-hand-side is one, which concludes the proof. □ □

Constraints (NACut) can be separated in polynomial time as follows: Given a fractional solution $(\tilde{x}, \tilde{y}, \tilde{y}^h)$ and a node w and layer h, consider all nodes v, such that $(v,w) \in A$, and calculate the sum $\sum_{v:(v,w) \in A} \min\{\tilde{x}_{vw}, \tilde{y}_v^{h-1}\}$. If the resulting sum is smaller than the LP-value of y_w^h, a violated inequality is obtained.

There is an interesting connection between the constraints (NACut) and constraints (g-HLink), which can be derived as follows. For a fixed v' and w, let $r \in R := \{r_{v'w} = 0; r_{vw} = 1, \forall v \neq v'\}$. Consider the aggregation of (NACut) over all $2 \leq h \leq H$. We obtain

$$\sum_{(v,w) \in A, v \neq v', r} (H-1) x_{vw} + \sum_{h \in H_2} y_{v'}^{h-1} \geq \sum_{h \in H_2} y_w^h.$$

Since the right hand side can be at most one, we can downlift the coefficient $(H-1)$ on the left hand side to one. Using equation (Root-Link), we obtain $x_{rw} + \sum_{(v,w)\in A, v\neq v', r} x_{vw} + \sum_{h\in H_2} y_{v'}^{h-1} \geq \sum_{h\in H_1} y_w^h$.This can be further rewritten using equations (Indegr) and (NH-Link) to get $x_{rw} + \sum_{(v,w)\in A, v'\neq v, r} x_{vw} + \sum_{h\in H_2} y_{v'}^{h-1} \geq \sum_{(v,w)\in A} x_{vw}$. Canceling out the x-variables on both sides, we arrive at

$$\sum_{h\in H_2} y_{v'}^{h-1} \geq x_{v'w},$$

which is an inequality of the family (g-HLink).

Flow-Conservation Constraints The so-called flow-conservation constraints (FlowC), which have been shown to strengthen the directed (prize-collecting) Steiner tree cut formulation, (see, e.g., [21, 24]) are easily seen to be valid for NODEHOP.

$$x(\delta^-(v)) \leq x(\delta^+(v)), \quad \forall v \in S \tag{FlowC}$$

The constraints ensure (in the x-space) that no node in S can be a leaf node in an solution. They can be generalized in a version involving y^h-variables in a similar fashion to (NACut).

Theorem 7. *Let F be the family of functions $F = 2^A$ and $f \in F$, $v \in S$ and $2 \leq h \leq H$. Then the* hop-flow-conservation-inequalities

$$\sum_{(v,w)\in A} \left(f_{vw} x_{vw} + (1 - f_{vw}) y_v^h \right) \geq y_v^{h-1} \tag{HFlowC}$$

are valid for NODEHOP.

Proof. Suppose there exists a feasible solution, where y_v^{h-1} is one, i.e., node v lies on layer $h-1$, and the left-hand-side is zero. However, since the node v is a Steiner node, there always exists an optimal solution, where v is no leaf node. Thus, there must be an arc (v,w) to some node v lying on layer h in the solution, thus both y_w^h and (v,w) must be one. One of these variables is on the left-hand-side of constraint (NACut), and thus the left-hand-side is one, which concludes the proof. □ □

3 The Solution Framework

We have implemented a branch-and-cut algorithm based on our model, using the state-of-the-art commercial solver CPLEX 12.6. Before the branch-and-cut algorithm gets started, a preprocessing phase, as presented in Section 3.1 is performed. Moreover, a primal heuristic (described in Section 3.2) is also part of our solution framework. Branching-priorities and details of the separation routines are described in Section 3.3. The selection of the valid inequalities to include in our framework is discussed in Section 4 together with the computational results.

3.1 Preprocessing

The aim of the preprocessing phase is to remove nodes, arcs and hop-indexed variables, which cannot be in an optimal solution. Moreover, the information gained in this phase also allows the lifting of some of the inequalities of the model. Let $dist(u,v)$ be the distance between two nodes u and v, this distance can be calculated with the help of a breadth-first-search (BFS). Moreover, let $dist(v,T) = \min_{w\in T} dist(v,w)$ be the distance between v and a closest node from the terminal set T. Note that these distances are calculated on a digraph, since the *directed root cost test* may remove some arcs in one direction. Some of the following results use the fact, that there always exists an optimal solution for STPRBH (and other Steiner tree problems), where no Steiner node $v \in S$ is a leaf.

Shrinking the Size of the Model Note that our model is defined on a digraph, while the problem is defined on an undirected graph. In our preprocessing, we first work on the undirected graph, and try to remove as much edges/nodes as possible, before we transform it into the directed graph on which our model is defined. The digraph is then further preprocessed. The preprocessing is comprised of the following tests:

- *directed root cost test:* If arcs (v,w) and (r,w) exist, and it holds that $c_{rw} \leq c_{vw}$, arc (v,w) can be removed, since w can always be connected to the root node. This test has been described in [19] for the hop-constrained spanning tree problem. A variant of this test, denoted by *undirected root cost test*, can be done before the transformation into a directed graph: If edges $\{v,w\}$, $\{r,v\}$ and $\{r,w\}$ exist, and it holds that $c_{rv} \leq c_{vw}$ and $c_{rw} \leq c_{vw}$, edge $\{v,w\}$ can be removed.

- *degree-one test:* This is a classical test from Steiner tree literature (see, e.g., [9]), every Steiner node with degree-one can be removed. Note that the *degree-two test*, which combines two edges into one, is not possible in our setting due to the op-constraints.

- *start/end-layer test:* Obviously, all variables y_v^h with $h < dist(r,v)$, where r is the root node, can be removed. For a similar approach, see also [23]. By definition of $dist(v,T)$, if $v \in S$, we must cross at least $dist(v,T)-1$ layers in order to reach a node in T from v. It follows that all variables y_v^h with $h > H - dist(v,T)$ can be removed. Consequently, all variables nodes v with $dist(r,v) + dist(v,T) > H$, can be removed. Also, all arcs (v,w), with $dist(r,v) \geq H - dist(w,T)$ can be removed.

The preprocessing starts with the *undirected root cost test*, followed by the *degree-one test*. Then the graph is transformed in a digraph and the *directed root cost test* is applied followed by the *start/end-layer test*. Finally, the *degree-one test* is the done again, since the *start/end-layer test* may remove some nodes, which could allow the *degree-one test* to also remove some additional nodes. Note that due to the *directed root cost test*, we can end up with $|dist(v,T) - dist(w,T)| > 1$ for two nodes $v,w \in S$ with an edge $\{v,w\}$ in the original graph since one of the arcs (v,w) or (w,v) may be removed.

Lifting Inequalities based on Preprocessing Preprocessing can be used to lift some of the valid inequalities. We demonstrate this on the family (HLink). First, observe that for any arc (v,w) with $dist(v,T) > 0$, e.g., for all $v \in S$, the lifting by adding y_v^H, which has been done to obtain (HLink) from (HLink-) does not have any effect, since the *start/end-layer test* removed the y_v^H variable. However, due to the information gained by the *dist*-calculation, some other variables y_v^h could be added to the left-hand-side. Suppose we are given an arc (v,w) with $dist(w,T) > dist(v,T)$. Let $h^*(w) = H - dist(w,T)$, i.e., h^* is the last layer, where w can lie, define $h^*(v)$ analogously. Thus, whenever, y_v^h is in the solution for some $h \geq h^*(w)$, the arc (v,w) cannot be taken. Following the same argumentation as for the lifting of (HLink-) to (HLink) by adding y_v^H, the lifting works by adding $\sum_{k=h^*(w)}^{h^*(v)} y_v^k$ to the left-hand-side of (HLink-). The resulting lifted inequalities (l-HLink) are denoted by *lifted hop-link inequalities*.

$$\sum_{k=h^*(w)}^{h^*(v)} y_v^k + y_v^{h-1} + x_{vw} \leq y_v + y_w^h, \quad \forall (v,w) \in A, v \neq r, dist(r,v) + 1 \leq h \leq h^*(v,w). \tag{l-HLink}$$

where $h^*(v,w) = min(h^*(v), h^*(w))$. Recall that in case the interval for which the constraint would be defined is empty, the *start/end-layer test* has removed the variable x_{vw}. Note that the sum can only go to $h^*(v)$, since the other y_v^h variables have been removed. Some caution must be taken, when the constraints (l-HLink) for a given arc only remain for one layer due to preprocessing. In this case, the validity of the lifting does not hold anymore, since it is based on the condition that at least two constraints (l-HLink) for an arc exist in the model. A (lifted) version of constraints (HEnd-) denoted by *lifted hop-end inequalities* (see (l-HEnd)) needs to be added in this case.

$$\sum_{k=h^*(w)}^{h^*(v)} y_v^k + x_{vw} \leq y_v \tag{l-HEnd}$$

Observe that for $dist(w,T) = 0$, i.e., $w \in T$, the original version of (HLink) remains, since the sum boils down to y_v^H. Lifted versions of the families (g-HLink),(o2AHLink), (e2AHLink)follow immediately by using the same ideas, i.e., the summation only needs to be over the range, where no y_v^h, y_w^h variables have been removed by preprocessing (respectively in the separation of these inequalities, the removed variables can be viewed as fixed to zero , and are always preferred to be ”taken” in the separated inequality). This latter view can also be applied to the separation of inequalities (NACut) and (HFlowC).

In addition to this lifting above, for all flow-balance inequalities (FlowC), where both the indegree and the outdegree of v is one, the inequality can be replaced by equality.

3.2 Primal Heuristic

Our primal heuristic is a modification of the improved version Prim-I [1] of the well-known Prim-based Steiner tree heuristic [28]. The heuristic works similar to Prim's minimum spanning tree algorithm [26], which starts with some node (the root node r, in our case) and then greedily grows the solution tree Sol by adding the node $v \notin Sol$, with minimum connection cost to Sol, i.e., the minimum cost edge $e = argmin\{c_{e=vs} : (v,s) : v \in V \setminus Sol, s \in Sol\}$, until all nodes are added. In the Steiner tree case, the solution Sol is grown by greedily adding terminal nodes $t \notin Sol$, with minimum connection cost, the connection cost is now not the cost of a single edge, but the cost from Sol to the terminal. When adding the chosen terminal to Sol, all the nodes on the paths are also added to Sol. We modified the algorithm Prim-I for the STPRBH, by taking the hop-limit and the budget into account. This can be easily achieved, since Prim-I works similar to Dijsktra's shortest path algorithm: Whenever an arc is going to be considered as part of a shortest path to a terminal, we check, if the hop-constraint is still fulfilled after adding the arc (note that for this check, the value $H - dist(v,T)$ can be used, instead of the hop-limit), if not, we ignore the connection offered by the arc. The budget-constraint is checked, whenever a terminal is added, if it would be violated, we of course do not add the terminal. Moreover, if the LP-value $\tilde{y}_v$ of a terminal-variable $v \in T$ is smaller than 0.001, we consider the terminal as Steiner node in the algorithm. When using this algorithm as primal heuristic, we set the arc weights to $\bar{c}_a = c_a(1 - \tilde{x}_a)$, where $\tilde{x}_a$ is the current LP-value of variable x_a. We have also experimented to take the information offered by LP-values $\tilde{y}_v^h$ into account for the arc weights, but in general this produced worse results. A simple local search consisting of exchange of leaf nodes is done at the end as improvement procedure. The algorithm is also used as starting heuristic, in this case, the original arc weights c_a are used.

The primal heuristic is put in the `heuristic callback` of CPLEX, which gets called after each LP at the root node and at the end of each node in the branch-and-bound tree. Moreover, we also call it in the `lazy constraint callback` of CPLEX, this callback gets called, whenever CPLEX encounters an integer solution. Such integer solutions can be produced by interal heuristics of CPLEX (which we explicitly turned on). In case that we do not add all inequalities of NODEHOP in the beginning, but separate them on the fly, these solutions produced by CPLEX may violate some of the not-yet added constraints, but can be repaired to feasible solutions (e.g., if not all inequalities (l-HLink) are already added, CPLEX can set some y^h to a wrong value). We thus aim to repair such a solution with a call to our primal heuristic. Moreover, we also try a simpler repairing procedure, which just consists of setting the right values for the y^h variables (this of course will not work, if the heuristic solution violated the hop-constraint). If we are successful in repairing, we store the solution and add it to CPLEX at the next call of the `heuristic callback`.

3.3 Further Enhancements

Branching Priorities The branching priorities are set as following: Each variable y_v is assigned priority $p_v + 1 + H$, each variable y_v^h gets priority $H - h$ and arc variables are assigned priority zero. This setting is chosen, since we conjecture that the most important decision in the STPRBH is to decide, which nodes, especially nodes with positive revenue, are in the solution. Moreover, if a node v lies on a layer near the root node, it is likely to greater influence the structure of the solution, than v lying on a layer near H.

Details of the Separation Routines The presented families of inequalities are all of a large size, some of them are even of exponential size, thus it is not practicable to add (all of) them in the beginning of the branch-and-cut algorithm, but separate them on-the-fly, when they are violated by an LP-solution. The separation of (the lifted versions of) inequalities (g-HLink), (o2AHLink), (e2AHLink), (NACut) has already be discussed above, inequalities (l-HLink), (FlowC) and (HFlowC) can also be separated in polynomial time by inspection.

Inequalities (CCuts) are separated using a max-flow algorithm [5], when the LP-relaxation is fractional, and using a BFS when an integer solution is encountered. The max-flow separation is enhanced using *minimum-cardinality cuts*, and *nested cuts*, moreover, we only add *back-cuts*, i.e., the incoming cut in the terminal-component, when the separation gives back more than one potential cuts, see [21, 24] for more details. The terminals are permuted before separation, so that we do not always separate to the same terminal first, since using nested cuts changes the capacities for subsequent separations. Moreover, also in the fractional case, we start a BFS on all arcs with LP-value one, and only do the separation for terminals not reachable this way. Once we have finished the separation for a terminal, we also start a similar BFS from this terminal, and all terminals found this way are also not considered for separation. Additionally, before adding a cut, we check if the nodes outside the component, to which the cut is incoming, could provide enough revenue to construct a better solution than the current incumbent. If not, we replace the y-variable on the right-hand-side of the cut to add with one, since any optimal solution must take a node from this component.

4 Computational Results

The algorithm is implemented in C++ and compiled using `g++4.9.2` with option `O3`. The framework OGDF [25] is used for graph-data-structures and CPLEX 12.6 is used as ILP-solver. The dual simplex algorithm with steepest edge pricing was chosen to solve the LP-relaxations. The computational results are obtained using a single core of an Intel E5-2670v2 with 2.5GHz and 64GB RAM. We used a time limit of 1 000 seconds for our testruns.

4.1 Instances

We tested our algorithm on the instances provided at the 11th DIMACS implementation challenge on Steiner trees, available at [8]. These instances have been proposed by [7] and [14]. Both are based on the graphs from the sets B and C of the Steiner tree problem graphs of OR-lib [2]. The transformation into STPRBH-instances is done as follows:

- terminal nodes from the STP are used as profitable nodes by associating a random profit to it (see Table 1)
- the budget B is determined as $\sum_{e\in E} c_e/b$, where b is a given divisor
- a hop-limit H is given

Using this transformation, the following set of instances have been created in [7] and [14] (see Table 1).

Table 1: Instances from the DIMACS-homepage. Instances of the upper group have been proposed by [7], the remaining ones by [14].

graphs	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert T\rvert$	p	b	H	number of inst.
B1-B18	50-100	63-200	9-50	[1-100]	5, 20	3, 6, 9, 12	144
C01-C05	500	625	5-250	[1,10], [1,100]	10, 30	5, 15, 25	60
C06-C10	500	1000	5-250	[1,10], [1,100]	20, 50	5, 15, 25	60
C11-C15	500	2500	5-250	[1,10], [1,100]	10, 100	5, 15, 25	60
C16-C20	500	12500	5-250	[1,10], [1,100]	100, 200	5, 15, 25	60
C16	500	12500	5	[1,10], [1,100]	10000	5, 15, 25	6
C17	500	12500	10	[1,10], [1,100]	5000	5, 15, 25	6
C18-C20	500	12500	83-250	[1,10], [1,100]	1000	5, 15, 25	18

Following [14], the instances can be grouped in five categories according to their difficulty.

- Group G1 contains all instances based on set B. They have been solved to optimality by exact algorithms [7, 27] (some of them also by [22]). The size of this group is 144.
- Group G2 contain the instances based on C01-C05. They have also been solved to optimality by exact algorithms [7, 27] (again, some of them also by [22]). The size of this group is 60.

The remaining three groups, based on larger (denser) graphs than G1 and G2, have only been tackled with heuristics so far.

- Group G3 contains instances proposed by [7], for which the trivial bound (namely the sum $\sum_{v \in T} p_v$) is the optimal solution value. For all G3 instances, heuristics from [7, 13, 14] were able to establish corresponding feasible (and, thus, optimal) solutions connecting all terminals within the given budget. The size of this group is 124.

- Group G4 contains the remaining instances proposed by [7]. For these instances, the optimal solutions are unknown. The size of this group is 56.

- Group G5 contains the instances proposed by [14]. The optimal solutions for these instances are unknown. The size of this group is 30.

4.2 Studying the Influence of the Valid Inequalities

In this section, we analyze the influence of the valid inequalities to the performance of the branch-and-cut approach. As a testbed for this analysis, we focus on G5 which is the most difficult group of instances. We consider the value of the LP-relaxation at the root node and the running time needed to obtain this value, as two main indicators for the usefulness of proposed valid inequalities.

We compare the following settings:

- *basic*: This is our initial model that consists of constraints (Indegr), (NH-Link), (Root-Link), (Budget), (FlowC), (GSEC2) and a constraint, that the root must have at least one outgoing arc (this is a special case of constraints (CCuts)). Inequalities (l-HLink) and (l-HEnd) are separated on the fly by enumeration, since preliminary runs showed that including all of them in the initial model slows down the performance.

- *cut*: This is *basic* enlarged by (CCuts) that are dynamically separated, and

- *nacut*: This is *basic*, enlarged by (CCuts) and (NACut), both of them being dynamically separated.

For these three settings, Figures 3 and 4 show performance profiles considering the LP-gaps and the running time at the root node of the branch-and-cut tree, respectively. The LP-gaps are calculated with respect to the optimal/best known solution.

It can be seen from Figure (3) that both (CCuts) and (NACut) improve the quality of LP-relaxation bounds. Comparing the running times needed to solve the root node relaxation, it turns out that there is a significant trade-off between the separation time required by (NACut), and the quality of attained bounds. More precisely, for 7 out of 30 instances, calculation of the LP-bounds at the root node has been aborted due to the imposed time limit. Nevertheless, the obtained bounds were always better than those achieved by *basic* and *cut*.

We have also investigated the influence of inequalities (g-HLink), (o2AHLink), (e2AHLink), (FlowC), (HFlowC) in this manner, however, we do not report detailed results in the above figures for sake of readability. It turned out that inequalities (g-HLink), (o2AHLink) and (e2AHLink) all help to improve the quality of LP-bounds. However, the improvement is rather marginal, at a a very high cost of increasing the overall running time. On the other hand, inequalities (FlowC) and (HFlowC), did not help in improving the LP-gaps.

In these experiments, we often observed a tailing-off effect, i.e., subsequent separation and resolving of LPs did only marginally improve the gaps after a certain number of iterations. We thus implemented a tailing-off control for the cut-loop. If $ub_{prev} - ub_{cur} < \rho$, where ub_{prev} is the bound obtained from the previous LP-relaxation, ub_{cur} the bound obtained by the current one, and ρ is a given parameter, we skip the separation routines and resort to branching. Figures 5 and 6 report the performance profiles concerning the obtained root relaxation gaps and associated running times for *basic*, *cut* and *nacut* with $\rho = 0.0001$, respectively.

Compared to the settings without the tailing-off control the gaps do not change too much. On the other hand, the time needed to solve the root relaxation drastically decreases. This may be explained by the fact that only the y-variables appear in the objective function, and the continuous addition of violated inequalities mainly influences the values of the x and y^h variables (without significantly changing the values of y-variables). A more sophisticated cut-loop scheme, like, e.g., in-out separation considered in [4, 12], could theoretically further improve the performance, however, we did not investigate this further, since the current tailing-off control already worked very well within the branch-and-cut algorithm, as it is demonstrated in the next section.

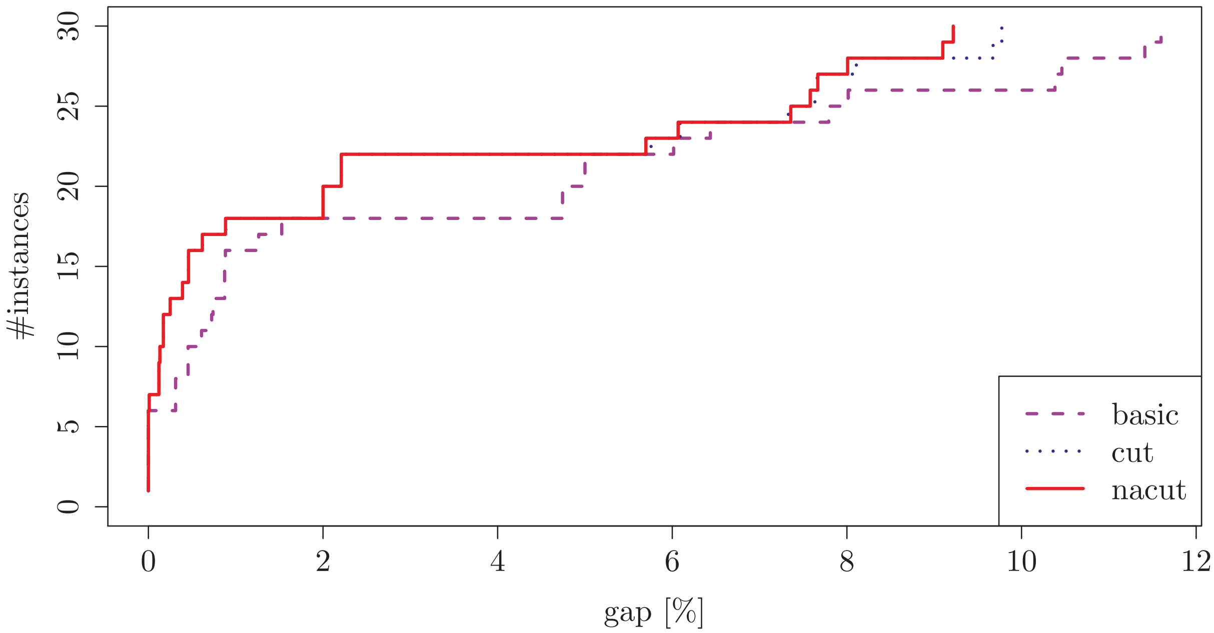


Figure 3: Root relaxation gaps for three different settings.

4.3 Main Results

For our main runs, setting *nacut* was chosen, with the tailing-off parameter ρ set to 0.0001. The global upper bound of the branch-and-cut tree is taken as ub_{prev} for the tailing-off test. Note that (NACut) are added to CPLEX using the `purgeable` option—this option allows CPLEX to remove constraints, if it deems them as not helpful. The following general purpose cuts of CPLEX have been set to one (moderate generation of cuts): `fractional`, `zero-half`, `cover`, all the other cuts are left at the default parameter.

In this section we concentrate on 86 instances of groups G4 and G5, for which the optimal solution values were unknown prior to this work. The results for group G4 are given in Table 2 and for group G5 in Table 3. Tables for groups G2 and G3 are given in the appendix (Tables 4 to 7). Note that our approach solves all instances from G2 and G3 to optimality, most of them already at the root node. Only a handful of instances from group G2 requires more than 10 seconds of computing time (but not more than 43 seconds), while for G3, the longest computing time is below 3 seconds.

Each table reports the obtained solution value ($sol.\ val$), which is shown in bold, if we have been able to prove optimality. The obtained global upper bound (UB) is also given (note that for the given instances, all costs/prizes are integers, thus we used $UB - sol.\ val < 1$ as stopping criterion). In addition, the gap after the timelimit is provided [$Gap\%$], as well as the root relaxation gap [$RGap\%$]. These gaps are given with respect to the best found solution value. If we have $UB - sol.\ val < 1$, **opt** is written instead. Note that this does not mean that optimality is proven at the root node, since the optimal solution may have not been found yet. On the other hand, CPLEX in some cases is able to use problem-specific information to prove optimality even if the root relaxation gap is greater than one. Moreover, due to repeated presolving and potential variable fixing and general purpose cuts of CPLEX, the root relaxation gaps can be different to the results reported in the previous section, where we looked at pure LPs. The time ($t[s]$) needed to prove optimality is also reported. If we were not able to prove optimality within our time limit of 1 000 seconds, the corresponding entry in the table is "-". The entry $tbest[s]$ contains the time when the best solution has been found and *nodes* gives the number of nodes in the branch-and-cut tree.

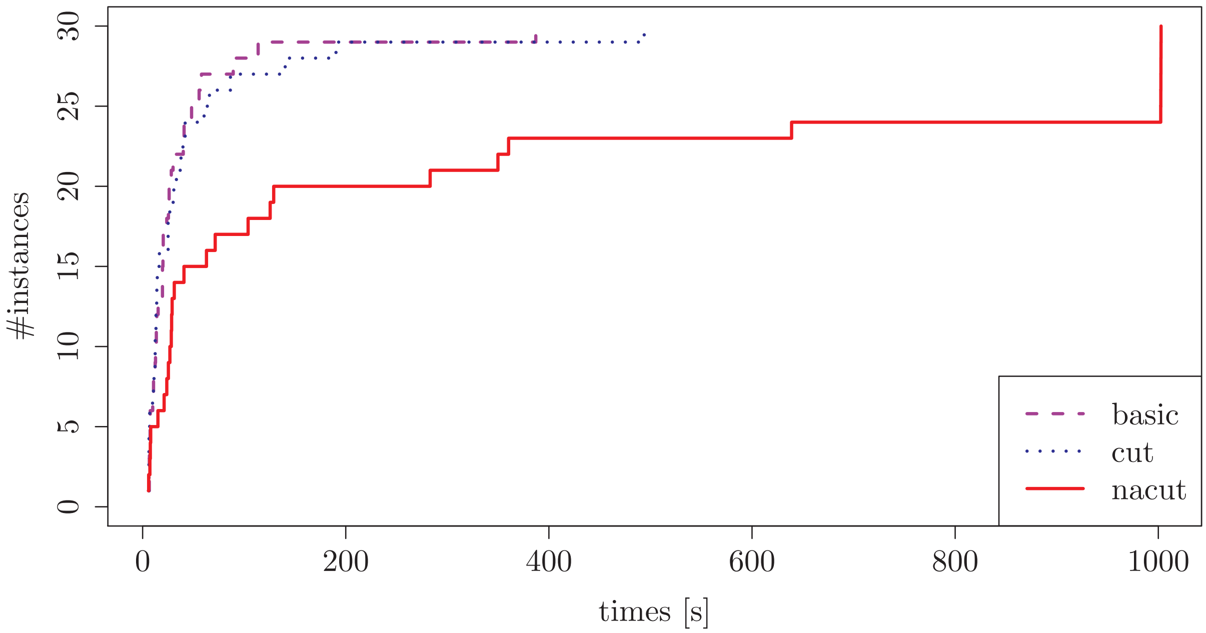


Figure 4: Time to solve the root relaxation for three different settings.

For instance group G4, we observe that only four of the 56 instances remain unsolved. Interestingly, these instances all have a hop-limit of 15 and a budget-divisor of 20. About half of the instances from this group can be solved within the root node, and (aside from the unsolved ones) only five instances need more than 60 seconds of computing time.

For instance group G5, also four instances remain unsolved — in contrast to group G4, three of the unsolved instances now have a hop-limit of 5, and only one has a hop-limit of 15. Again, about half of the instances from the group can be solved to optimality at the root node and (aside from the unsolved ones) only four instances need more than 100 seconds.

To summarize, out of 86 previously unsolved instances, only eight remain. As mentioned before, larger hop-limits are one of main bottlenecks for the exact methods considered in previous literature. The obtained results clearly demonstrate that our new approach deals very well with larger hop-limit, as we have been able to solve all instances from literature with a (largest considered) hop-limit of 25 to proven optimality.

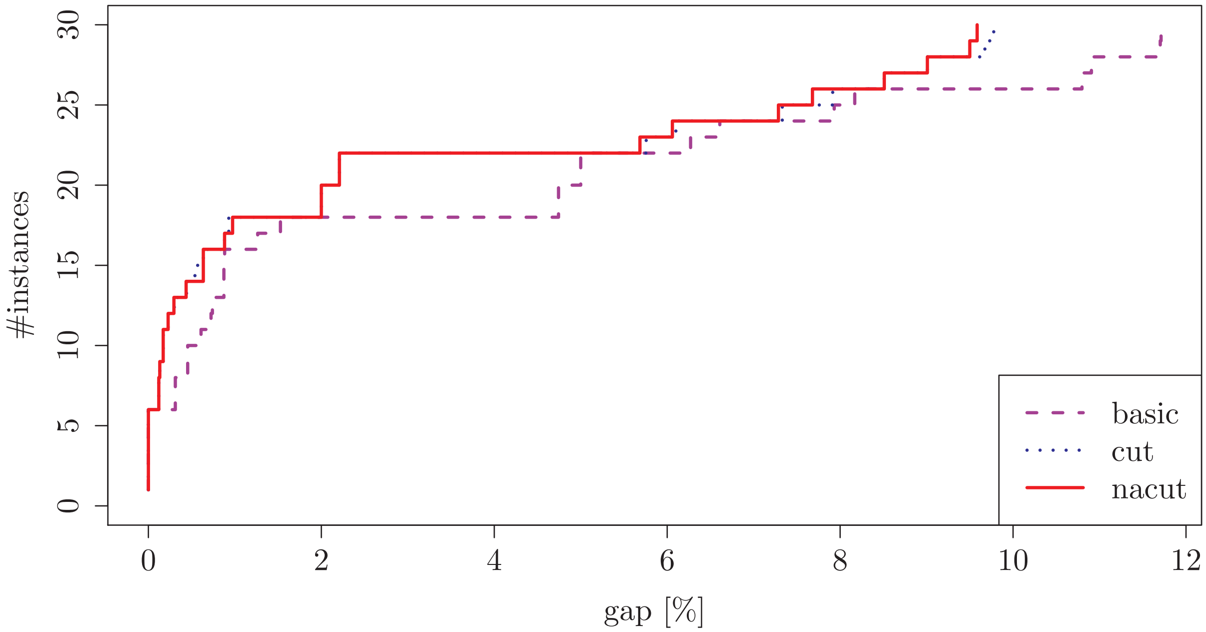


Figure 5: Root relaxation gaps for three different settings with tailing-off control.

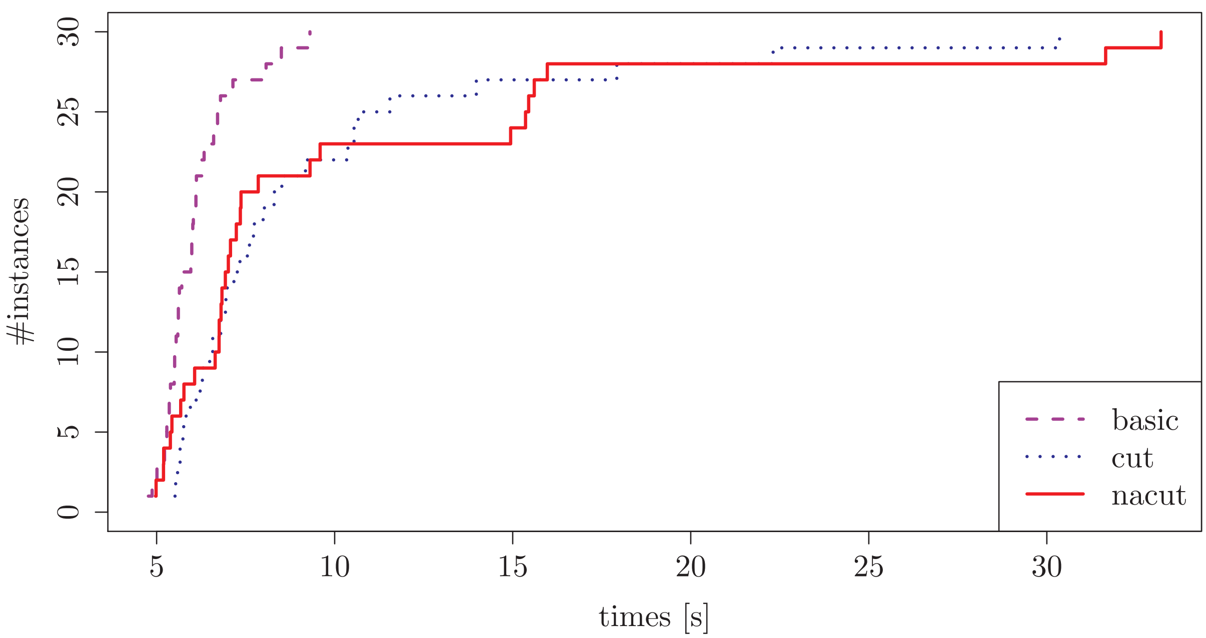


Figure 6: Time (in seconds) needed to solve the root relaxation for three different settings with tailing-off control.

Table 2: Results for group G4 of previously unsolved instances

inst	budget	hop	sol. val	UB	Gap [%]	RGap [%]	t [s]	tbest [s]	nodes
C08-10	20	5	**230**	230.00	**opt**	**opt**	0.04	0.03	0
C08-10	50	5	**116**	116.00	**opt**	**opt**	0.08	0.06	0
C08-10	20	15	**331**	331.00	**opt**	0.38	18.5	18.14	19
C08-10	50	15	**171**	171.00	**opt**	0.95	17.05	16.65	24
C08-10	20	25	**332**	332.00	**opt**	**opt**	3.46	2.89	0
C08-10	50	25	**172**	172.00	**opt**	**opt**	5.51	4.37	0
C08-100	20	5	**2380**	2380.00	**opt**	**opt**	0.06	0.02	0
C08-100	50	5	**1216**	1216.00	**opt**	0.21	0.13	0.03	2
C08-100	20	15	3431	3447.09	0.47	0.75	-	15.05	301
C08-100	50	15	**1776**	1776.00	**opt**	0.77	31.35	18.21	62
C08-100	20	25	**3455**	3455.00	**opt**	0.05	17.93	4.01	5
C08-100	50	25	**1792**	1792.00	**opt**	0.31	23.8	3.13	12
C09-10	20	5	**304**	304.00	**opt**	**opt**	0.71	0.21	0
C09-10	50	5	**149**	149.00	**opt**	0.67	0.68	0.17	2
C09-10	20	15	381	384.04	0.8	1.05	-	38.53	258
C09-10	50	15	**185**	185.00	**opt**	1.44	21.26	9.78	35
C09-10	20	25	**385**	385.00	**opt**	**opt**	11.07	10.37	2
C09-10	50	25	**187**	187.00	**opt**	**opt**	6.41	6.40	0
C09-100	20	5	**3133**	3133.00	**opt**	**opt**	0.59	0.19	0
C09-100	50	5	**1563**	1563.00	**opt**	**opt**	0.33	0.14	1
C09-100	20	15	**3945**	3945.00	**opt**	0.75	144.26	24.50	114
C09-100	50	15	**1906**	1906.00	**opt**	1.54	60.03	24.59	132
C09-100	20	25	**3974**	3974.00	**opt**	**opt**	15.54	10.91	1
C09-100	50	25	**1933**	1933.00	**opt**	**opt**	12.38	3.19	0
C10-10	20	5	**391**	391.00	**opt**	**opt**	0.1	0.08	0
C10-10	50	5	**185**	185.00	**opt**	**opt**	0.36	0.11	0
C10-10	20	15	573	580.59	1.32	1.51	-	427.87	123
C10-10	50	15	**257**	257.00	**opt**	**opt**	5.07	0.89	0
C10-10	20	25	**580**	580.00	**opt**	0.28	204.05	200.85	43
C10-10	50	25	**258**	258.00	**opt**	**opt**	5.26	4.07	0
C10-100	20	5	**4096**	4096.00	**opt**	**opt**	0.11	0.08	0
C10-100	50	5	**1940**	1940.00	**opt**	0.44	0.4	0.09	7
C10-100	20	15	5906	5983.91	1.32	1.46	-	92.20	116
C10-100	50	15	**2657**	2657.00	**opt**	0.53	22.31	4.77	14
C10-100	20	25	**5972**	5972.00	**opt**	0.36	206.1	25.59	193
C10-100	50	25	**2683**	2683.00	**opt**	**opt**	6.13	3.11	0
C13-10	100	5	**257**	257.00	**opt**	**opt**	5.35	5.35	0
C13-10	100	15	**319**	319.00	**opt**	0.63	21.62	17.47	16
C13-10	100	25	**319**	319.00	**opt**	0.63	57.38	43.81	22
C13-100	100	5	**2653**	2653.00	**opt**	**opt**	6.87	6.31	0
C13-100	100	15	**3312**	3312.00	**opt**	0.14	18.21	13.47	5
C13-100	100	25	**3317**	3317.00	**opt**	**opt**	33.43	33.42	0
C14-10	100	5	**373**	373.00	**opt**	**opt**	3.07	2.58	0
C14-10	100	25	**404**	404.00	**opt**	**opt**	7.15	7.14	0
C14-100	100	5	**3887**	3887.00	**opt**	**opt**	5.88	3.56	0
C14-100	20	15	**6566**	6566.00	**opt**	**opt**	0.41	0.05	0
C14-100	100	15	**4205**	4205.00	**opt**	**opt**	1.26	1.26	0
C14-100	100	25	**4205**	4205.00	**opt**	**opt**	4.31	4.30	0
C15-10	20	5	**1248**	1248.00	**opt**	**opt**	45.71	45.61	27
C15-10	100	5	**480**	480.00	**opt**	**opt**	3.5	3.33	0
C15-10	100	15	**568**	568.00	**opt**	0.35	63.19	61.94	52
C15-10	100	25	**569**	569.00	**opt**	**opt**	12.93	4.36	0
C15-100	20	5	**12533**	12533.00	**opt**	**opt**	35.42	32.21	17
C15-100	100	5	**5000**	5000.00	**opt**	**opt**	2.62	2.19	0
C15-100	100	15	**5889**	5889.00	**opt**	0.32	223.32	171.03	275
C15-100	100	25	**5905**	5905.00	**opt**	0.01	55.51	28.54	3

Table 3: Results for group G5 of previously unsolved instances

inst	budget	hop	sol. val	UB	Gap [%]	RGap [%]	t [s]	tbest [s]	nodes
C16-10	10000	5	**19**	19.00	**opt**	**opt**	5.27	0.16	0
C16-10	10000	15	**19**	19.00	**opt**	**opt**	7.68	7.67	0
C16-10	10000	25	**19**	19.00	**opt**	**opt**	17.39	0.19	0
C16-100	10000	5	**203**	203.00	**opt**	**opt**	5.4	0.17	0
C16-100	10000	15	**203**	203.00	**opt**	**opt**	8.14	8.13	0
C16-100	10000	25	**203**	203.00	**opt**	**opt**	18.17	0.19	0
C17-10	5000	5	**47**	47.00	**opt**	6.38	11.71	5.72	5
C17-10	5000	15	**50**	50.00	**opt**	**opt**	9.5	7.27	0
C17-10	5000	25	**50**	50.00	**opt**	**opt**	17.97	13.43	0
C17-100	5000	5	**481**	481.00	**opt**	3.33	22.58	0.15	7
C17-100	5000	15	**513**	513.00	**opt**	**opt**	13.66	7.16	0
C17-100	5000	25	**513**	513.00	**opt**	**opt**	22.17	16.08	0
C18-10	1000	5	318	322.54	1.43	1.69	-	123.47	51
C18-10	1000	15	**341**	341.00	**opt**	0.41	38.98	35.78	11
C18-10	1000	25	**341**	341.00	**opt**	0.37	89.76	63.70	7
C18-100	1000	5	3320	3366.71	1.41	1.5	-	73.77	50
C18-100	1000	15	**3552**	3552.00	**opt**	0.49	268.94	65.22	106
C18-100	1000	25	**3557**	3557.00	**opt**	0.25	244.38	232.40	16
C19-10	1000	5	**404**	404.00	**opt**	**opt**	49.76	43.32	0
C19-10	1000	15	**428**	428.00	**opt**	**opt**	12.73	7.75	0
C19-10	1000	25	**428**	428.00	**opt**	0.04	96.05	65.21	1
C19-100	1000	5	**4179**	4179.00	**opt**	0.32	81.67	31.00	25
C19-100	1000	15	**4435**	4435.00	**opt**	**opt**	25.5	20.42	0
C19-100	1000	25	**4435**	4435.00	**opt**	0.03	61.65	57.37	2
C20-10	1000	5	**460**	460.00	**opt**	0.76	305.45	71.42	37
C20-10	1000	15	502	506.20	0.84	0.9	-	50.21	102
C20-10	1000	25	**506**	506.00	**opt**	**opt**	33.45	25.32	0
C20-100	1000	5	**4768**	4768.00	**opt**	0.65	537.88	347.55	50
C20-100	1000	15	5222	5250.20	0.54	0.6	-	604.62	131
C20-100	1000	25	**5256**	5256.00	**opt**	**opt**	51.58	51.56	0

5 Conclusion and Outlook

The power of layered graphs has been recently demonstrated for many problems, including hop- and diameter-constrained spanning trees [19], hop-constrained connected facility location [23], or for problems that involve more general hop- or diameter-constraints (see, e.g., [16], [17]).

In this paper, we propose a new extended formulation based on a layered graph for hop-constrained spanning/Steiner tree problems. Our formulation follows a "thinning out" idea proposed in [10, 11]: instead of using variables associated with arcs of the layered graph, our new model projects them out and relies only on variables associated to the nodes of the layered graph. Thus, the resulting MIP formulation is considerably smaller than the ones considered in previous literature, which allows us to tackle instances based on larger graphs and/or hop-limits.

We apply the new model to solve the Steiner tree problem with revenues, budget and hop-constraints (STPRBH), which has been part of the DIMACS challenge [8]. A branch-and-cut approach based on our model allows us to significantly improve results from the available literature. Previous to our study, 86 out of 414 available instances have been unsolved. We prove the optimality for all except eight out of these 414 instances, often within seconds. For these remaining eight instances, we improve the best known solutions.

We consider the following topics as important directions for the future research:

- First, considering a theoretical comparison with other models for hop-constrained problems. A (potential) connection of our model to a disaggregation of MTZ-constraints (that have been recently revisited in [3]) is particularly interesting. Besides, it would be important to conduct a theoretical and computational comparison of the node-based versus the arc-based layered graph approach for the hop-constrained spanning/Steiner tree problem.
- Second, it is worth mentioning that diameter-constrained spanning/Steiner tree problems can also be solved using our new modeling approach. It remains an open question how the proposed model relates with the recent formulation derived in the natural space of edge variables (see [18]), and with the arc-based layered graph formulation studied in [19].
- Finally, we believe that broader applications involving hop- and diameter-constrained trees (see above), especially problems with large-scale instances, might significantly benefit from the proposed "thinning out" approach.

Acknowledgements

The research was supported by the Austrian Research Fund (FWF, Project P 26755-N19)

References

[1] de Aragão, M.P., Werneck, R.F.: On the implementation of MST-based heuristics for the Steiner problem in graphs. In: Algorithm Engineering and Experiments, pp. 1–15. Springer (2002)

[2] Beasley, J.E.: OR-Library: distributing test problems by electronic mail. Journal of the operational research society pp. 1069–1072 (1990)

[3] Bektaş, T., Gouveia, L.: Requiem for the miller–tucker–zemlin subtour elimination constraints? European Journal of Operational Research **236**(3), 820–832 (2014)

[4] Ben-Ameur, W., Neto, J.: Acceleration of cutting-plane and column generation algorithms: Applications to network design. Networks **49**(1), 3–17 (2007)

[5] Cherkassky, B.V., Goldberg, A.V.: On implementing the push—relabel method for the maximum flow problem. Algorithmica **19**(4), 390–410 (1997)

[6] Costa, A.M., Cordeau, J.F., Laporte, G.: Fast heuristics for the Steiner tree problem with revenues, budget and hop constraints. European Journal of Operational Research **190**(1), 68–78 (2008)

[7] Costa, A.M., Cordeau, J.F., Laporte, G.: Models and branch-and-cut algorithms for the Steiner tree problem with revenues, budget and hop constraints. Networks **53**(2), 141–159 (2009)

[8] DIMACS: 11th DIMACS Implementation Challenge in Collaboration with ICERM: Steiner Tree Problems (2014). `http://dimacs11.cs.princeton.edu/home.html`

[9] Duin, C.W., Volgenant, A.: Reduction tests for the Steiner problem in graphs. Networks **19**(5), 549–567 (1989). DOI 10.1002/net.3230190506

[10] Fischetti, M., Leitner, M., Ljubić, I., Luipersbeck, M., Monaci, M., Resch, M., Salvagnin, D.: Thinning out Steiner trees: A node-based model for uniform edge costs (2015). Submited

[11] Fischetti, M., Ljubić, I., Sinnl, M.: Thinning out facilities: a Benders decomposition approach for the uncapacitated facility location problem with separable convex costs (2015). Submitted

[12] Fischetti, M., Salvagnin, D.: An in-out approach to disjunctive optimization. In: A. Lodi, M. Milano, P. Toth (eds.) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, *Lecture Notes in Computer Science*, vol. 6140, pp. 136–140. Springer Berlin Heidelberg (2010)

[13] Fu, Z.H., Hao, J.K.: Breakout local search for the Steiner tree problem with revenue, budget and hop constraints. European Journal of Operational Research **232**(1), 209–220 (2014)

[14] Fu, Z.H., Hao, J.K.: Dynamic programming driven memetic search for the steiner tree problem with revenues, budget, and hop constraints. INFORMS Journal on Computing **27**(2), 221–237 (2015)

[15] Gouveia, L.: Using Hop-Indexed Models For Constrained Spanning and Steiner Tree Models. In: B. Sansò, P. Soriano (eds.) Telecommunications Network Planning, Centre for Research on Transportation, pp. 21–32. Springer US (1999)

[16] Gouveia, L., Leitner, M., Ljubić, I.: The two-level diameter constrained spanning tree problem. Mathematical Programming pp. 1–30 (2012)

[17] Gouveia, L., Leitner, M., Ljubić, I.: Hop constrained Steiner trees with multiple root nodes. European Journal of Operational Research **236**(1), 100–112 (2014)

[18] Gouveia, L., Leitner, M., Ljubić, I.: A polyhedral study of the diameter constrained minimum spanning tree problem (2015). Submited

[19] Gouveia, L., Simonetti, L., Uchoa, E.: Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs. Mathematical Programming **128**, 123–148 (2011)

[20] Hwang, F., Richards, D.S.: Steiner tree problems. Networks **22**(1), 55–89 (1992)

[21] Koch, T., Martin, A.: Solving Steiner tree problems in graphs to optimality. Networks **32**, 207–232 (1998)

[22] Layeb, S.B., Hajri, I., Haouari, M.: Solving the Steiner tree problem with revenues, budget and hop constraints to optimality. In: Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on, pp. 1–4. IEEE (2013)

[23] Ljubić, I., Gollowitzer, S.: Layered graph approaches to the hop constrained connected facility location problem. INFORMS Journal on Computing **25**(2), 256–270 (2013)

[24] Ljubić, I., Weiskircher, R., Pferschy, U., Klau, G.W., Mutzel, P., Fischetti, M.: An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. Mathematical Programming **105**(2-3), 427–449 (2006)

[25] OGDF: The Open Graph Drawing Framework. `http://www.ogdf.net/doku.php`

[26] Prim, R.C.: Shortest connection networks and some generalizations. Bell system technical journal **36**(6), 1389–1401 (1957)

[27] Sinnl, M.: Branch-and-price for the Steiner tree problem with revenues, budget and hop constraints. Master's thesis, Vienna University of Technology (2011)

[28] Takahashi, H., Matsuyama, A.: An approximate solution for the Steiner problem in graphs. Math. Japonica **24**(6), 573–577 (1980)

[29] Winter, P.: Steiner problem in networks: a survey. Networks **17**(2), 129–167 (1987)

Appendix: Detailed Results for previously solved instances based on set C

Table 4: Results for group G2 of instances previously solved to optimality by other exact approaches

inst	budget	hop	sol. val	UB	Gap [%]	RGap [%]	t [s]	tbest [s]	nodes
C01-10	10	5	**8**	8.00	**opt**	**opt**	0.01	0.00	0
C01-10	30	5	**8**	8.00	**opt**	**opt**	0.01	0.00	0
C01-10	10	15	**27**	27.00	**opt**	**opt**	0.07	0.02	0
C01-10	30	15	**27**	27.00	**opt**	**opt**	0.07	0.02	0
C01-10	10	25	**27**	27.00	**opt**	**opt**	0.14	0.02	0
C01-10	30	25	**27**	27.00	**opt**	**opt**	0.15	0.03	0
C01-100	10	5	**71**	71.00	**opt**	**opt**	0.01	0.00	0
C01-100	30	5	**71**	71.00	**opt**	**opt**	0.01	0.01	0
C01-100	10	15	**274**	274.00	**opt**	**opt**	0.06	0.01	0
C01-100	30	15	**274**	274.00	**opt**	**opt**	0.07	0.02	0
C01-100	10	25	**274**	274.00	**opt**	**opt**	0.15	0.02	0
C01-100	30	25	**274**	274.00	**opt**	**opt**	0.15	0.03	0
C02-10	10	5	**32**	32.00	**opt**	**opt**	0.01	0.00	0
C02-10	30	5	**32**	32.00	**opt**	**opt**	0.01	0.00	0
C02-10	10	15	**59**	59.00	**opt**	**opt**	0.08	0.02	0
C02-10	30	15	**53**	53.00	**opt**	**opt**	0.47	0.47	0
C02-10	10	25	**59**	59.00	**opt**	**opt**	0.15	0.02	0
C02-10	30	25	**53**	53.00	**opt**	**opt**	0.84	0.84	0
C02-100	10	5	**328**	328.00	**opt**	**opt**	0.03	0.02	0
C02-100	30	5	**328**	328.00	**opt**	**opt**	0.01	0.00	0
C02-100	10	15	**604**	604.00	**opt**	**opt**	0.09	0.02	0
C02-100	30	15	**546**	546.00	**opt**	**opt**	0.51	0.51	0
C02-100	10	25	**604**	604.00	**opt**	**opt**	0.15	0.02	0
C02-100	30	25	**546**	546.00	**opt**	**opt**	0.80	0.80	0
C03-10	10	5	**151**	151.00	**opt**	**opt**	0.01	0.01	0
C03-10	30	5	**95**	95.00	**opt**	**opt**	0.02	0.01	0
C03-10	10	15	**289**	289.00	**opt**	0.27	4.43	4.28	15
C03-10	30	15	**129**	129.00	**opt**	**opt**	0.82	0.31	0
C03-10	10	25	**289**	289.00	**opt**	**opt**	3.12	1.70	0
C03-10	30	25	**129**	129.00	**opt**	**opt**	1.90	1.10	0
C03-100	10	5	**1519**	1519.00	**opt**	**opt**	0.01	0.00	0
C03-100	30	5	**968**	968.00	**opt**	0.91	0.08	0.02	18
C03-100	10	15	**2971**	2971.00	**opt**	0.39	22.17	3.88	117
C03-100	30	15	**1343**	1343.00	**opt**	**opt**	0.89	0.16	0
C03-100	10	25	**2979**	2979.00	**opt**	0.34	21.77	3.08	62
C03-100	30	25	**1343**	1343.00	**opt**	**opt**	3.45	0.38	0
C04-10	10	5	**115**	115.00	**opt**	**opt**	0.01	0.00	0
C04-10	30	5	**84**	84.00	**opt**	**opt**	0.01	0.01	0
C04-10	10	15	**336**	336.00	**opt**	1.51	16.09	10.14	51
C04-10	30	15	**134**	134.00	**opt**	2.13	3.00	2.86	7
C04-10	10	25	**341**	341.00	**opt**	**opt**	3.06	1.96	0
C04-10	30	25	**136**	136.00	**opt**	**opt**	3.09	1.96	0
C04-100	10	5	**1148**	1148.00	**opt**	**opt**	0.01	0.01	0
C04-100	30	5	**854**	854.00	**opt**	**opt**	0.01	0.01	0
C04-100	10	15	**3458**	3458.00	**opt**	0.55	26.88	16.24	111
C04-100	30	15	**1380**	1380.00	**opt**	1.42	7.21	3.13	51
C04-100	10	25	**3504**	3504.00	**opt**	**opt**	6.55	0.42	0
C04-100	30	25	**1396**	1396.00	**opt**	0.24	12.61	11.65	4
C05-10	10	5	**258**	258.00	**opt**	**opt**	0.01	0.01	0
C05-10	30	5	**154**	154.00	**opt**	**opt**	0.01	0.01	0
C05-10	10	15	**494**	494.00	**opt**	0.53	17.07	17.03	27
C05-10	30	15	**182**	182.00	**opt**	1.4	9.26	8.20	18
C05-10	10	25	**495**	495.00	**opt**	0.37	22.70	21.01	18
C05-10	30	25	**183**	183.00	**opt**	0.84	7.71	6.22	2
C05-100	10	5	**2600**	2600.00	**opt**	**opt**	0.01	0.00	0
C05-100	30	5	**1584**	1584.00	**opt**	**opt**	0.01	0.00	0
C05-100	10	15	**5032**	5032.00	**opt**	0.47	20.07	16.96	37
C05-100	30	15	**1857**	1857.00	**opt**	0.91	7.50	6.14	35
C05-100	10	25	**5044**	5044.00	**opt**	0.23	28.86	27.49	45
C05-100	30	25	**1860**	1860.00	**opt**	0.76	16.37	12.61	34

Table 5: Results for group G3 of instances previously solved to optimality by heuristics, 1/3

inst	budget	hop	sol. val	UB	Gap [%]	RGap [%]	t [s]	tbest [s]	nodes
C06-10	20	5	**27**	27.00	**opt**	**opt**	0.01	0.01	0
C06-10	50	5	**27**	27.00	**opt**	**opt**	0.01	0.01	0
C06-10	20	15	**27**	27.00	**opt**	**opt**	0.16	0.03	0
C06-10	50	15	**27**	27.00	**opt**	**opt**	0.14	0.03	0
C06-10	20	25	**27**	27.00	**opt**	**opt**	0.33	0.04	0
C06-10	50	25	**27**	27.00	**opt**	**opt**	0.39	0.04	0
C06-100	20	5	**274**	274.00	**opt**	**opt**	0.01	0.01	0
C06-100	50	5	**274**	274.00	**opt**	**opt**	0.01	0.01	0
C06-100	20	15	**274**	274.00	**opt**	**opt**	0.16	0.03	0
C06-100	50	15	**274**	274.00	**opt**	**opt**	0.17	0.04	0
C06-100	20	25	**274**	274.00	**opt**	**opt**	0.34	0.04	0
C06-100	50	25	**274**	274.00	**opt**	**opt**	0.38	0.04	0
C07-10	20	5	**49**	49.00	**opt**	**opt**	0.01	0.01	0
C07-10	50	5	**49**	49.00	**opt**	**opt**	0.01	0.01	0
C07-10	20	15	**59**	59.00	**opt**	**opt**	0.18	0.03	0
C07-10	50	15	**59**	59.00	**opt**	**opt**	0.18	0.03	0
C07-10	20	25	**59**	59.00	**opt**	**opt**	0.40	0.04	0
C07-10	50	25	**59**	59.00	**opt**	**opt**	0.40	0.04	0
C07-100	20	5	**503**	503.00	**opt**	**opt**	0.01	0.01	0
C07-100	50	5	**503**	503.00	**opt**	**opt**	0.01	0.01	0
C07-100	20	15	**604**	604.00	**opt**	**opt**	0.15	0.03	0
C07-100	50	15	**604**	604.00	**opt**	**opt**	0.17	0.03	0
C07-100	20	25	**604**	604.00	**opt**	**opt**	0.34	0.04	0
C07-100	50	25	**604**	604.00	**opt**	**opt**	0.39	0.04	0
C11-10	20	5	**27**	27.00	**opt**	**opt**	0.04	0.02	0
C11-10	100	5	**27**	27.00	**opt**	**opt**	0.04	0.02	0
C11-10	20	15	**27**	27.00	**opt**	**opt**	0.37	0.05	0
C11-10	100	15	**27**	27.00	**opt**	**opt**	0.35	0.05	0
C11-10	20	25	**27**	27.00	**opt**	**opt**	0.69	0.06	0
C11-10	100	25	**27**	27.00	**opt**	**opt**	0.67	0.06	0
C11-100	20	5	**274**	274.00	**opt**	**opt**	0.04	0.01	0
C11-100	100	5	**274**	274.00	**opt**	**opt**	0.04	0.02	0
C11-100	20	15	**274**	274.00	**opt**	**opt**	0.33	0.05	0
C11-100	100	15	**274**	274.00	**opt**	**opt**	0.36	0.06	0
C11-100	20	25	**274**	274.00	**opt**	**opt**	0.65	0.06	0
C11-100	100	25	**274**	274.00	**opt**	**opt**	0.65	0.06	0
C12-10	20	5	**59**	59.00	**opt**	**opt**	0.05	0.02	0
C12-10	100	5	**59**	59.00	**opt**	**opt**	0.05	0.02	0
C12-10	20	15	**59**	59.00	**opt**	**opt**	0.34	0.05	0
C12-10	100	15	**59**	59.00	**opt**	**opt**	0.32	0.04	0
C12-10	20	25	**59**	59.00	**opt**	**opt**	0.63	0.06	0
C12-10	100	25	**59**	59.00	**opt**	**opt**	0.65	0.06	0
C12-100	20	5	**604**	604.00	**opt**	**opt**	0.05	0.02	0
C12-100	100	5	**604**	604.00	**opt**	**opt**	0.06	0.02	0
C12-100	20	15	**604**	604.00	**opt**	**opt**	0.37	0.05	0
C12-100	100	15	**604**	604.00	**opt**	**opt**	0.37	0.05	0
C12-100	20	25	**604**	604.00	**opt**	**opt**	0.71	0.07	0
C12-100	100	25	**604**	604.00	**opt**	**opt**	0.71	0.07	0

Table 6: Results for group G3 of instances previously solved to optimality by heuristics, 2/3

inst	budget	hop	sol. val	UB	Gap [%]	RGap [%]	t [s]	tbest [s]	nodes
C13-10	20	5	**439**	439.00	**opt**	**opt**	0.10	0.03	0
C13-10	20	15	**439**	439.00	**opt**	**opt**	0.38	0.05	0
C13-10	20	25	**439**	439.00	**opt**	**opt**	0.67	0.06	0
C13-100	20	5	**4463**	4463.00	**opt**	**opt**	0.11	0.03	0
C13-100	20	15	**4463**	4463.00	**opt**	**opt**	0.34	0.04	0
C13-100	20	25	**4463**	4463.00	**opt**	**opt**	0.72	0.06	0
C14-10	20	5	**648**	648.00	**opt**	**opt**	0.29	0.29	0
C14-10	20	15	**648**	648.00	**opt**	**opt**	0.39	0.05	0
C14-10	100	15	**404**	404.00	**opt**	**opt**	4.64	2.39	0
C14-10	20	25	**648**	648.00	**opt**	**opt**	0.67	0.06	0
C14-100	20	5	**6566**	6566.00	**opt**	**opt**	0.29	0.28	0
C14-100	20	25	**6566**	6566.00	**opt**	**opt**	0.68	0.06	0
C15-10	20	15	**1248**	1248.00	**opt**	**opt**	0.43	0.06	0
C15-10	20	25	**1248**	1248.00	**opt**	**opt**	0.78	0.07	0
C15-100	20	15	**12533**	12533.00	**opt**	**opt**	0.39	0.06	0
C15-100	20	25	**12533**	12533.00	**opt**	**opt**	0.75	0.07	0
C16-10	100	5	**27**	27.00	**opt**	**opt**	0.68	0.14	0
C16-10	200	5	**27**	27.00	**opt**	**opt**	0.70	0.15	0
C16-10	100	15	**27**	27.00	**opt**	**opt**	1.29	0.18	0
C16-10	200	15	**27**	27.00	**opt**	**opt**	1.32	0.16	0
C16-10	100	25	**27**	27.00	**opt**	**opt**	2.08	0.18	0
C16-10	200	25	**27**	27.00	**opt**	**opt**	2.05	0.17	0
C16-100	100	5	**274**	274.00	**opt**	**opt**	0.75	0.16	0
C16-100	200	5	**274**	274.00	**opt**	**opt**	0.71	0.15	0
C16-100	100	15	**274**	274.00	**opt**	**opt**	1.45	0.19	0
C16-100	200	15	**274**	274.00	**opt**	**opt**	1.34	0.16	0
C16-100	100	25	**274**	274.00	**opt**	**opt**	2.08	0.18	0
C16-100	200	25	**274**	274.00	**opt**	**opt**	2.14	0.19	0
C17-10	100	5	**59**	59.00	**opt**	**opt**	0.75	0.15	0
C17-10	200	5	**59**	59.00	**opt**	**opt**	0.72	0.17	0
C17-10	100	15	**59**	59.00	**opt**	**opt**	1.27	0.15	0
C17-10	200	15	**59**	59.00	**opt**	**opt**	1.27	0.15	0
C17-10	100	25	**59**	59.00	**opt**	**opt**	1.95	0.16	0
C17-10	200	25	**59**	59.00	**opt**	**opt**	2.06	0.17	0
C17-100	100	5	**604**	604.00	**opt**	**opt**	0.71	0.16	0
C17-100	200	5	**604**	604.00	**opt**	**opt**	0.73	0.15	0
C17-100	100	15	**604**	604.00	**opt**	**opt**	1.32	0.16	0
C17-100	200	15	**604**	604.00	**opt**	**opt**	1.39	0.17	0
C17-100	100	25	**604**	604.00	**opt**	**opt**	2.19	0.19	0
C17-100	200	25	**604**	604.00	**opt**	**opt**	1.96	0.16	0

Table 7: Results for group G3 of instances previously solved to optimality by heuristics, 3/3

inst	budget	hop	sol. val	UB	Gap [%]	RGap [%]	t [s]	tbest [s]	nodes
C18-10	100	5	**439**	439.00	**opt**	**opt**	0.83	0.16	0
C18-10	200	5	**439**	439.00	**opt**	**opt**	0.79	0.15	0
C18-10	100	15	**439**	439.00	**opt**	**opt**	1.37	0.16	0
C18-10	200	15	**439**	439.00	**opt**	**opt**	1.38	0.20	0
C18-10	100	25	**439**	439.00	**opt**	**opt**	2.15	0.17	0
C18-10	200	25	**439**	439.00	**opt**	**opt**	2.19	0.19	0
C18-100	100	5	**4463**	4463.00	**opt**	**opt**	0.71	0.14	0
C18-100	200	5	**4463**	4463.00	**opt**	**opt**	0.75	0.14	0
C18-100	100	15	**4463**	4463.00	**opt**	**opt**	1.34	0.15	0
C18-100	200	15	**4463**	4463.00	**opt**	**opt**	1.37	0.15	0
C18-100	100	25	**4463**	4463.00	**opt**	**opt**	2.17	0.18	0
C18-100	200	25	**4463**	4463.00	**opt**	**opt**	2.23	0.19	0
C19-10	100	5	**648**	648.00	**opt**	**opt**	0.72	0.14	0
C19-10	200	5	**648**	648.00	**opt**	**opt**	0.83	0.16	0
C19-10	100	15	**648**	648.00	**opt**	**opt**	1.39	0.16	0
C19-10	200	15	**648**	648.00	**opt**	**opt**	1.39	0.16	0
C19-10	100	25	**648**	648.00	**opt**	**opt**	2.21	0.19	0
C19-10	200	25	**648**	648.00	**opt**	**opt**	2.13	0.17	0
C19-100	100	5	**6566**	6566.00	**opt**	**opt**	0.73	0.15	0
C19-100	200	5	**6566**	6566.00	**opt**	**opt**	0.79	0.15	0
C19-100	100	15	**6566**	6566.00	**opt**	**opt**	1.42	0.16	0
C19-100	200	15	**6566**	6566.00	**opt**	**opt**	1.45	0.18	0
C19-100	100	25	**6566**	6566.00	**opt**	**opt**	2.30	0.18	0
C19-100	200	25	**6566**	6566.00	**opt**	**opt**	2.07	0.16	0
C20-10	100	5	**1248**	1248.00	**opt**	**opt**	0.78	0.15	0
C20-10	200	5	**1248**	1248.00	**opt**	**opt**	0.76	0.15	0
C20-10	100	15	**1248**	1248.00	**opt**	**opt**	1.39	0.17	0
C20-10	200	15	**1248**	1248.00	**opt**	**opt**	1.38	0.16	0
C20-10	100	25	**1248**	1248.00	**opt**	**opt**	2.24	0.20	0
C20-10	200	25	**1248**	1248.00	**opt**	**opt**	2.03	0.16	0
C20-100	100	5	**12533**	12533.00	**opt**	**opt**	0.83	0.16	0
C20-100	200	5	**12533**	12533.00	**opt**	**opt**	0.91	0.18	0
C20-100	100	15	**12533**	12533.00	**opt**	**opt**	1.51	0.18	0
C20-100	200	15	**12533**	12533.00	**opt**	**opt**	1.56	0.20	0
C20-100	100	25	**12533**	12533.00	**opt**	**opt**	2.31	0.20	0
C20-100	200	25	**12533**	12533.00	**opt**	**opt**	2.32	0.20	0