Math. Program., Ser. A (2011) 128:123–148
DOI 10.1007/s10107-009-0297-2

FULL LENGTH PAPER

Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs

Luis Gouveia · Luidi Simonetti · Eduardo Uchoa

Received: 27 October 2008 / Accepted: 15 June 2009 / Published online: 22 July 2009

Abstract The hop-constrained minimum spanning tree problem (HMSTP) is an NP-hard problem arising in the design of centralized telecommunication networks with quality of service constraints. We show that the HMSTP is equivalent to a Steiner tree problem (STP) in an appropriate layered graph. We prove that the directed cut model for the STP defined in the layered graph, dominates the best previously known models for the HMSTP. We also show that the Steiner directed cuts in the extended layered graph space can be viewed as being a stronger version of some previously known HMSTP cuts in the original design space. Moreover, we show that these strengthened cuts can be combined and projected into new families of cuts, including facet defining ones, in the original design space. We also adapt the proposed approach to the diameter-constrained minimum spanning tree problem (DMSTP). Computational results with a branch-and-cut algorithm show that the proposed method is significantly better than previously known methods on both problems.

L. Gouveia (✉)
Faculdade de Ciências da Universidade de Lisboa,
DEIO, CIO Bloco C/2, Campo Grande, 1749-016 Lisbon, Portugal
e-mail: legouveia@fc.ul.pt

L. Simonetti
Universidade Federal do Rio de Janeiro, PESC/COPPE, Bl. H, sala 319,
Rio de Janeiro 21941-972, Brazil
e-mail: luidi@cos.ufrj.br

E. Uchoa
Departamento de Engenharia de Produção, Universidade Federal Fluminense,
R. Passo da Pátria, 156, Bl.E sala 440, Niterói 24210-240, Brazil
e-mail: uchoa@producao.uff.br

Keywords Networks/graphs: tree algorithms · Integer programming: formulations · Cutting planes

Mathematics Subject Classification (2000) 90C11 Mixed integer programming · 90C27 Combinatorial optimization · 90C57 Polyhedral combinatorics, branch-and-bound, branch-and-cut

1 Introduction

The hop-constrained minimum spanning tree problem (HMSTP) is defined as follows: given a graph $G = (V, E)$ with node set $V = \{0, 1, \ldots, n\}$ and edge set E as well as a positive cost c_e associated with each edge e of E and a natural number H, we wish to find a spanning tree T of the graph with minimum total cost and such that the unique path from a specified root node, node 0, to any other node has no more than H hops (edges).

The HMSTP is NP-hard because it contains as a particular case (the case with $H = 2$) an NP-Hard version of the Simple Uncapacitated Facility Location problem (see [3,8]). Manyem and Stallmann [22] have shown that the HMSTP is not in APX, i.e., the class of problems for which it is possible to have polynomial time heuristics with a constant-factor approximation bound even if P $\neq$ NP. The HMSTP models the design of centralized telecommunication networks with quality of service constraints. The root node represents the site of a central processor and the remaining nodes represent terminals that are required to be linked to the central processor. The hop constraints guarantee a certain level of service with respect to some performance constraints such as availability and reliability (see [33]). Centralized terminal networks are also usually implemented with multidrop lines for connecting the terminals with the center. In such networks, node processing times dominate over link delays and fewer hops mean, in general, lower delays.

Lower bounding schemes for the HMSTP based on network flow models have been suggested in [9,10,14]. More recently, Dahl et al. [5] propose a model involving only natural design variables and an exponential sized set of so-called “jump” constraints and propose a lower bound based on an appropriate Lagrangian relaxation. The recent survey by Dahl et al. [6] summarizes these approaches. The reader is also referred to the survey by Kerivin and Mahjoub [18] on general network design problems with hop constraints and methods to solve them.

The models described in the previous papers view the HMSTP problem as defined in the original graph (although some extended models also use variables corresponding to arcs in auxiliary layered graphs associated to hop-constrained shortest path problems, one for each non-root node). In this paper we propose a modeling approach for the HMSTP that views the whole problem as defined in a single layered graph. We will show that the HMSTP is equivalent to a Steiner tree problem (STP) (see, for instance [17,20]) in that layered graph. Thus, any known model for this problem can be used to solve the HMSTP. Our computational results are taken from an adaptation of the efficient branch-and-cut method proposed in [25], and which is based on the well known directed cut model for the STP. We will also show that the linear programming relaxation bounds of the directed cut model defined in the layered graph are at least as good (strictly better for several instances) than the linear programming bounds of the best HMST models presented so far. We give some insight of what inequalities are implied by the linear programming relaxation of the new model that are not redundant in the linear programming relaxation of the previous ones. We provide a technique for combining inequalities from the new model in a such a way that they can be projected into new families of valid inequalities, including proven facet defining ones, in the original design space.

A related problem, the so-called diameter-constrained minimum spanning tree problem (DMSTP) is defined as follows: given a graph $G = (V, E)$ with node set $V = \{1, \ldots, n\}$ and edge set E as well as a positive cost c_e associated with each edge e of E and a natural number D, we wish to find a spanning tree T of the graph with minimum total cost and such that the unique path from any node i to any node j has no more than D hops (edges). Note that in this variation, we constrain the path between each pair of nodes while in the previous one, only the paths from the special node are constrained. Several approaches for the DMSTP (see, for instance [1,11,16,27]) have used the properties of tree centers in order to transform the DMSTP into special versions of the HMSTP. In a similar way, here we adapt the new HMTSP method for the DMSTP. Although the adaptation given for the situations with even diameter is taken in a straightforward manner from the literature, we propose a new transformation for the odd diameter case that permits us to make efficient use of the layered graph approach.

The HMSTP and the DMSTP are originally defined over undirected graphs. However, we will focus our presentation on so-called directed formulations which lead, in general, to models with a tighter linear programming bound (see, for instance [21]). To view the HMSTP as a directed problem, we define a directed graph $G = (V, A)$, such that for each edge $\{i, j\}$ in E, we have two arcs (i, j) and (j, i) in A with the same cost as the original edge. With respect to edges $(0, j)$, we consider only a single arc $(0, j)$. Now the problem looks for a hop-constrained spanning tree directed away from 0. A similar construct will be done for the DMSTP, since in the proposed approaches, a dummy central node is created and we can view the problem as determining a tree directed away from it.

The remainder of the paper is as follows. In Sect. 2 we describe the layered graph and the transformation of the HMSTP into a STP in the layered graph, providing a corresponding model. In Sect. 3 we make a brief survey of the best formulations presented in the literature and compare their linear programming bounds with the linear programming relaxation bounds of the new model. This section also presents new inequalities in the design space that are implied by the new model. In Sect. 4 we review the STP branch-and-cut method described in [25]. In Sect. 5 we describe an adaptation of the transformation and method for the related DMSTP, for both even or odd diameters. The computational experiments are given in Sect. 6. The last section presents some conclusions.

2 Transforming the HMSTP into a STP in a layered graph: Hop-Cut model

In this section we show how to transform the HMSTP into a directed STP in a layered graph. Consider a graph $G_L = (V_L, A_L)$ defined as follows:

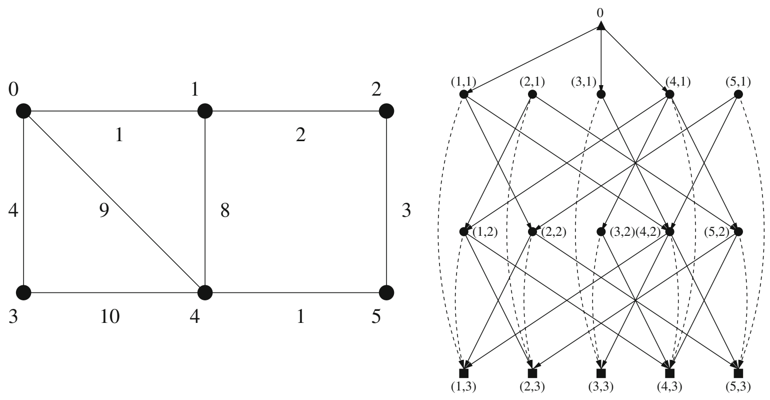

Fig. 1 Layered graph transformation: original graph of an instance with $H = 3$ on the *left-hand side* and the corresponding layered graph on the *right-hand side*

$$\begin{aligned}
&V_L = \{0\} \cup \{(i,h) : 1 \le h \le H \quad \text{and} \quad i \in V\backslash\{0\}\} \quad \text{and}\\
&A_L = A_0 \cup A_1 \cup A_2 \quad \text{where}\\
&A_0 = \{((0),(j,1)) : (0,j) \in A\},\\
&A_1 = \{((i,h),(j,h+1)) : (i,j) \in A, i \ne 0, 1 \le h \le H-1\} \quad \text{and}\\
&A_2 = \{((i,h),(i,H)) : i \in V\backslash\{0\}, 1 \le h \le H-1\}.
\end{aligned}$$

The root node is 0 and the required node set R is defined as $\{(i,H) : i \in V\backslash\{0\}\}$. The cost of the arcs in the sets A_0 and A_1 are equal to the costs of the corresponding arcs in the original graph and the cost of arcs in A_2 is equal to 0. Note that G_L is built by levels with the nodes of the original graph replicated H times. A node (i,h) in the layered graph is associated to node i being in depth h in the original graph (i.e., the path from node 0 to node i contains h arcs). Figure 1 illustrates on its right-hand side the layered graph corresponding to the graph shown on the left-hand side, for an instance with $H = 3$. The root node is depicted as a triangle, nodes of R are depicted as squares, the arcs in A_2 are dashed.

Consider the arcs in any Steiner tree in G_L. Removing the arcs in A_2 and ignoring the indices h of the remaining arcs, one obtains a spanning tree with depth less or equal than H in the original graph and with the same cost. Conversely, every hop constrained spanning tree in the original graph corresponds to a Steiner tree with the same cost in the layered graph. One example of such pair of solutions is given in Fig. 2.

With this transformation, we can use any model for the STP to provide a valid model for the HMSTP. We choose the well known directed cut formulation given in [20] (see also [2]). Define the following non-negative variables: (i) X_{0j}^1 for each arc $(0,(j,1))$ in A_0, (ii) X_{ij}^h for each arc $((i,h-1),(j,h))$ in A_1, and (iii) X_{jj}^h to each arc $((j,h-1),(j,H))$ in A_2. Let $[V_L\backslash S, S]$ denote a cut in the layered graph such that $0 \notin S$ and $S \cap R \ne \{\emptyset\}$ and let $X[V_L \setminus S, S]$ denote the sum of the X_{ij}^h variables associated to arcs on that cut. Also define binary variables x_{ij}, for each (i,j) in A, corresponding to each arc in the original graph. The following model is nothing else than the directed cut Steiner model rewritten for the layered graph:

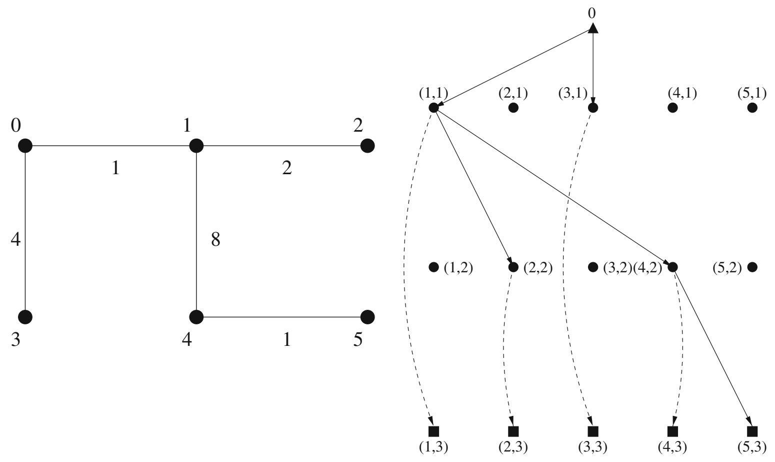

Fig. 2 Optimal HMST in the original graph (cost 16) and its corresponding Steiner tree in G_L $(H = 3)$

Model Hop-Cut

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1}$$

$$\text{s.t.} X[V_L \backslash S, S] \geq 1, \quad \forall S \subset V_L;\; S \cap R \neq \{\emptyset\};\; 0 \notin S \tag{2}$$

$$x_{0j} = X^1_{0j}, \quad \forall j \in V\backslash\{0\} \tag{3}$$

$$x_{ij} = \sum_{h=2}^{H} X^h_{ij}, \quad \forall (i,j) \in A;\; i \neq 0. \tag{4}$$

$$X \geq 0, \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i,j) \in A. \tag{6}$$

Constraints (2) are directed cut constraints and state that if some required nodes are in S, then any feasible solution for the problem must contain one arc, at least, in the cut $[V_L \setminus S, S]$. There is an exponential number of such constraints, but they can be efficiently separated by the min-cut algorithm. Constraints (3) and (4) just link the X and the x variables.

3 Comparing the model Hop-Cut with previous models

The model in the literature that has been providing the best results was proposed by Gouveia [10]. For each $k \in V\backslash\{0\}$, define non-negative flow variables y^{1k}_{0j} indicating whether arc $(0, j) \in A$ is in the path from 0 to k in the original graph. For each $k \in V\backslash\{0\}, h = 2, \ldots, H$, define flow variables y^{hk}_{ij} indicating whether arc $(i, j) \in A$, $i \neq 0$, is the hth arc in the path from 0 to node k in the original graph. The Hop-MCF model can be written as follows:

Model Hop-MCF

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{7}$$

$$\text{s.t.} \sum_{i\in V\setminus\{j\}} x_{ij} = 1, \quad j \in V\setminus\{0\} \tag{8}$$

$$\sum_{j\in V\setminus\{0\}} y_{0j}^{1k} = 1, \quad k \in V\setminus\{0\} \tag{9}$$

$$y_{0i}^{1k} - \sum_{j\in V\setminus\{0,i\}} y_{ij}^{2k} = 0, \quad i,k \in V\setminus\{0\}; i \neq k \tag{10}$$

$$\sum_{j\in V\setminus\{0,i,k\}} y_{ji}^{hk} - \sum_{j\in V\setminus\{0,i\}} y_{ij}^{(h+1)k} = 0, \quad i,k \in V\setminus\{0\}; i \neq k; h = 2,\ldots,H-1 \tag{11}$$

$$y_{0j}^{1k} \leq x_{0j}, \quad (0,j) \in A; k \in V\setminus\{0\} \tag{12}$$

$$\sum_{h=2}^{H} y_{ij}^{hk} \leq x_{ij}, \quad (i,j) \in A; i \neq 0; k \in V\setminus\{0\}; \tag{13}$$

$$y \geq 0, \tag{14}$$

$$x_{ij} \in \{0,1\}, \quad (i,j) \in A. \tag{15}$$

Constraints (9–11) define, for each $k \in V\setminus\{0\}$, an unitary flow of commodity k from 0 to the set of nodes $\{(k,h)|h = 1,\ldots,H\}$ over a layered graph similar to G_L. Constraints (12–13) link the x and the y variables. In spite of having variables defined over the same layered graph, model Hop-Cut is stronger than Hop-MCF. The proof of the following theorem can be found in [15].

Theorem 1 *The projection of the set of feasible solutions from the linear relaxation of the Hop-Cut model into the space of the x variables is contained in the set of feasible solutions from the linear programming relaxation of the Hop-MCF model projected into the same space.*

The solution shown in Fig. 3 is an extreme point of the feasible set of the linear relaxation of Hop-MCF, for an instance with $n = 4$ and $H = 3$. On the left-hand side it depicts the x variables with value 0.5. The corresponding y variables for $k = 3$ and 4, also with value 0.5, are depicted on the right-hand side of the same figure. This x solution is not feasible for the Hop-Cut model. This example gives some intuition why the new model has a stronger linear relaxation. In the model Hop-MCF, an arc with positive value (such as arc $(2,4)$) allows positive flows in different layers in different y paths (layer 3 in the path to node 4 and layer 2 in the path to node 3). A similar situation does not happen in model Hop-Cut because there is a single set of auxiliary variables X for all required nodes k. A detailed explanation of this fact can also be found in [15], where a network flow version of the Hop-Cut model is shown to be equivalent to a disaggregated version of the Hop-MCF model.

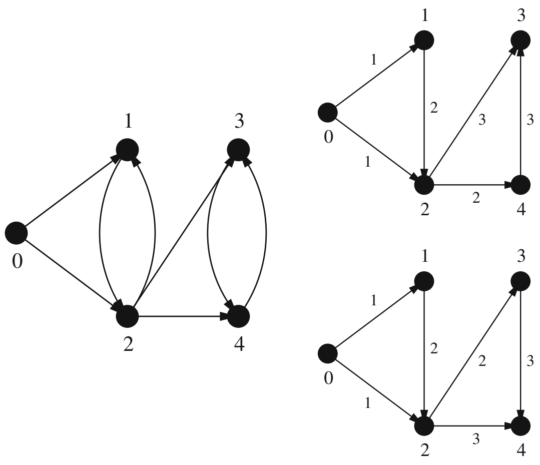

Fig. 3 Fractional solution of Hop-MCF model over the x variables (on the *left-hand side*) and corresponding y variables for $k = 3$ and 4 (on the *right-hand side*), labels in the arcs indicate hop index h

One interesting point is to know whether there exist formulations using only the x variables and which are equivalent to the previous extended formulations. A partial answer to this question was given in [6]. Let $S_0, S_1, \ldots, S_{H+1}$ be node-disjoint sets defining a partition of the node set V such that each subset is nonempty, and that $S_0 = \{0\}$. We then define $J = J(S_0, S_1, \ldots, S_{H+1}) = \cup_{[i+1<j]}[S_i, S_j]$ where $[S_i, S_j]$ is the set of arcs $\{(u, v) \in A : u \in S_i, v \in S_j\}$. We call such set J a H-jump. Let J_H denote the set of all H-jumps. Consider, now, the following set of inequalities, the jump inequalities,

$$\sum_{(p,q)\in J} x_{pq} \geq 1, \quad J \in J_H. \tag{16}$$

Loosely speaking, constraints (16) are valid because any feasible path from S_0 to S_{H+1} needs to make a jump somewhere from a node set S_i to one of the sets $S_{i+2}, \ldots, S_{H+1}$. Those constraints were first proposed by Dahl [4]. The following formulation was shown to be valid for the HMSTP by [6]:

Model Jump

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{17}$$

$$\text{s.t.} \quad (9), (16)$$

$$\sum_{(p,q)\in[V\setminus S,S]} x_{pq} \geq 1, \quad S \subseteq V;\ 0 \notin S \tag{18}$$

$$\sum_{(p,q)\in J} x_{pq} \geq 1, \quad J \in J_H. \tag{19}$$

Note that constraints (18) are the usual cut constraints. Dahl et al. [6] have shown that the projection of the set of feasible solutions of the linear programming relaxation of Hop-MCF into the space of the x variables is equal to the set of feasible solutions of the linear programming relaxation of the Jump model when $H = 3$ and that strict inclusion occurs when $H > 3$ (when $H = 2$ the HMSTP is equivalent to a Simple Facility Location Problem and both models are dominated by (8), (15), and $x_{ij} \leq x_{0i}$, $(i, j) \in A, i \neq 0$). In spite of being weaker than Hop-MCF when $H > 3$, the formulation Jump may be worthy of investigation for its compactness (in terms of variables).

In the following, we assume that the original graph $G = (V, E)$ is complete and therefore the directed arc set A contains all possible arcs, except those converging into node 0 (that is, $|A| = n^2$). Also assume that $H \geq 3$ and $n \geq H + 1$. Define $R(n, H)$ as the set of solutions of the formulation Jump, each solution is a 0–1 vector in R^{n^2}. Define $P(n, H)$ as the convex hull of $R(n, H)$. Dahl et al. [5] have already proved some properties on this polyhedron, namely that: (i) the dimension of $P(n, H)$ is $n^2 - n$ (constraints (8) are the only equalities containing it); (ii) constraint jump (16) corresponding to $J = J(S_0, S_1, \dots, S_{H+1})$ defines a facet of $P(n, H)$ if and only if $|S_{H+1}| = 1$. Furthermore, it is quite easy to show that a cut constraint (18) corresponding to $[V\backslash S, S]$ defines a facet of $P(n, H)$ if and only if $|S| > 1$. In the next subsection we further explore the polyhedron $P(n, H)$.

3.1 New inequalities in the x space by projection

As noted before, the Hop-Cut model can yield bounds strictly better than the bounds of the Hop-MCF and Jump models, even for $H = 3$. It would be interesting to have some insight of what inequalities are implied by their linear programming relaxation and that are not redundant in the linear programming relaxation of the Hop-MCF and Jump formulations. More precisely, we will show that the linear programming relaxation of the Hop-Cut model implies new inequalities of the form

$$\sum_{k=1}^{K_j} \sum_{(p,q)\in J_k} x_{pq} + \sum_{k=1}^{K_c} \sum_{(p,q)\in[V\backslash S_k, S_k]} x_{pq} \geq |K_j| + |K_c| + \sum_{i,j} \gamma_{ij} x_{ij}, \qquad (20)$$

where J_k, $k = 1, \dots, K_j$, are jump sets, $[V\backslash S_k, S_k]$, $k = 1, \dots, K_c$, are cut sets and γ_{ij} are nonnegative lifting coefficients.

The procedure to obtain such inequalities has two steps. First, we show that the linear programming relaxation of the Hop-Cut formulation implies the following lifted versions (in the (x,X) space) of the jump and the cut inequalities (given in a generic form):

$$\sum_{(p,q)\in J} x_{pq} \geq 1 + \sum_{i,j,h} \alpha_{ij}^h X_{ij}^h, \qquad (21)$$

$$\sum_{(p,q)\in[V\backslash S,S]} x_{pq} \geq 1 + \sum_{i,j,h} \beta_{ij}^h X_{ij}^h, \qquad (22)$$

where J and $[V\backslash S, S]$ are jump and cut sets respectively, and α_{ij}^h and β_{ij}^h are nonnegative lifting coefficients. Second, these two sets of lifted inequalities suggest that if we add several constraints (21) or add some constraints (21) with some constraints (22), we might be able to use the linking constraints (3) and (4) on some of the variables on the right side of the resulting inequality in order to obtain a constraint of the form (20). As an example, suppose that for a given arc (i, j), $i \neq 0$, cut $[V\backslash S, S]$ and jump J we are able to obtain the following two lifted inequalities

$$\sum_{(p,q)\in J} x_{pq} \geq 1 + \sum_{h\in H_1} X_{ij}^h, \tag{23}$$

$$\sum_{(p,q)\in[V\backslash S,S]} x_{pq} \geq 1 + \sum_{h\in H_2} X_{ij}^h, \tag{24}$$

where $H_1 \cup H_2 = \{2, \ldots, H\}$. Adding up the two inequalities and using (4) gives

$$\sum_{(p,q)\in J} x_{pq} + \sum_{(p,q)\in[V\backslash S,S]} x_{pq} \geq 2 + x_{ij}, \tag{25}$$

which is stronger than the inequality obtained by adding the original jump and cut inequality.

3.1.1 Lifted jump and lifted cut inequalities in the space of the x and X variables

First, we show how to obtain one general version of a lifted jump inequality (21). For a fixed hop parameter H, let us consider a partition $S_0, S_1, \ldots, S_{H+1}$, $S_0 = \{0\}$, and let us consider the layered graph partitioned in a similar manner, that is, set S_i^h corresponds to the node set S_i in the layer h of the layered graph (for $i = 1, \ldots, H+1$ and $h = 1, \ldots, H$). Now, consider a cut $[A, B]$ in the layered graph such that

$$B = \bigcup_{h=1,\ldots,H} \left(\bigcup_{i=h+1,\ldots,H+1} S_i^h \right). \tag{26}$$

Figure 4 gives an example of such a cut in the layered graph for an instance with $H = 3$ and $n = 4$. Now, by adding adequate trivial inequalities $X_{ij}^h \geq 0$ to the left-hand side of the cut inequality and by using the linking constraints (3) and (4) we obtain the following lifted jump inequality:

$$\sum_{(p,q)\in J} x_{pq} \geq 1 + \sum_{j=1}^{H-2} \sum_{i=j+2}^{H} \sum_{h=i}^{H} X^h[S_j, S_i] + \sum_{j=2}^{H-1} \sum_{i=j+2}^{H+1} \sum_{h=2}^{j} X^h[S_j, S_i],$$
$$J = J(S_0, S_1, \ldots, S_{H+1}). \tag{27}$$

Note that the X_{ij}^h variables used on the expression on the right-hand side of the inequality correspond to the trivial inequalities that have been added to the original

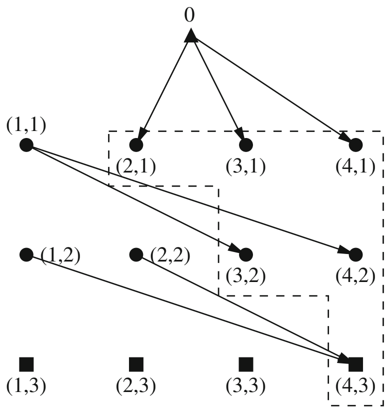

Fig. 4 Cut in the layered graph corresponding to a lifting of a jump inequality in the original graph

cut inequality. Note also, that the arcs associated to these variables either are inside the set A (arcs associated to variables in the first summation term in the right-hand side of the inequality) or inside the set B (arcs associated to variables in the second summation in right-hand side) of the cut.

As an example consider an instance with $H = 3$ and $n = 4$, and the partition $J = J(\{0\}, \{1\}, \{2\}, \{3\}, \{4\})$. Then, the previous inequality becomes

$$\sum_{(p,q)\in J} x_{pq} = x_{02} + x_{03} + x_{04} + x_{13} + x_{14} + x_{24} \geq 1 + X_{13}^{3} + X_{24}^{2}, \tag{28}$$

as shown in Fig. 4.

With respect to the lifted cut inequality (22), let S, $1 < |S| < n$, be a set in the original graph not containing the root and consider the cut set $C = [V\backslash S, S]$. For any vertex $k \in S$, there is a lifted cut inequality

$$\sum_{(p,q)\in C} x_{pq} \geq 1 + \sum_{i\in V\backslash S, i\neq 0} \sum_{j\in S\backslash\{k\}} X_{ij}^{H}. \tag{29}$$

Consider, again, $n = 4$ and $H = 3$. When $S = \{3, 4\}$ and $k = 3$, we obtain

$$\sum_{(p,q)\in C} x_{pq} = x_{03} + x_{04} + x_{13} + x_{14} + x_{23} + x_{24} \geq 1 + X_{14}^{3} + X_{24}^{3}, \tag{30}$$

as shown in Fig. 5.

3.1.2 Generating new inequalities in the x space

First, we show how to generate a constraint that can be seen as a stronger version of an inequality that is obtained by adding a lifted jump inequality (27) with a lifted cut inequality (29). The main observation to generate the new inequality is to note that in

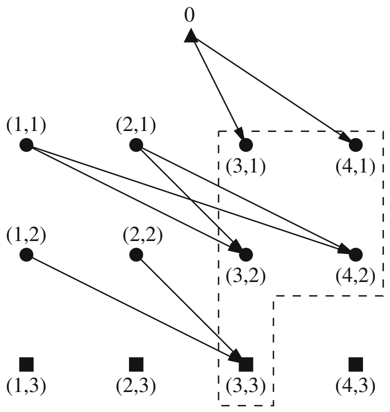

Fig. 5 Cut in the layered graph corresponding to a lifting of a cut inequality in the original graph

the lifted variables of (29) the hop index h is always equal to H and that the second summation in the right-hand side of (27) contains the expression

$$\sum_{h=2}^{H-1} X^h[S_{H-1}, S_{H+1}].$$

Thus, if for a fixed hop parameter H, we consider a partition $J = \{S_0, S_1, \ldots, S_{H+1}\}$ with $S_0 = \{0\}$, for the lifted jump inequality and, for the lifted cut inequality, we consider a set $S = S_H \cup S_{H+1}$, and add the two inequalities, we obtain

$$\sum_{(p,q)\in J} x_{pq} + \sum_{(p,q)\in C} x_{pq} \geq 2 + \sum_{h=2}^{H} X_h[S_{H-1}, S_{H+1}] + \text{“extra terms”}. \quad (31)$$

By discarding the extra terms and using (4) on the summation, we obtain the *jump-cut inequality*:

$$\sum_{(p,q)\in J} x_{pq} + \sum_{(p,q)\in C} x_{pq} \geq 2 + x[S_{H-1}, S_{H+1}]. \quad (32)$$

For an instance with $n = 4$, $H = 3$, $J = J[\{0\}, \{1\}, \{2\}, \{3\}, \{4\}]$, $C = [\{0, 1, 2\}, \{3, 4\}]$ the previous expression becomes

$$x_{02} + 2x_{03} + 2x_{04} + 2x_{13} + 2x_{14} + x_{24} + x_{23} \geq 2, \quad (33)$$

which is exactly the inequality obtained from the sum of the two lifted inequalities (28) and (30). It can be seen that this inequality cuts the fractional solution shown in Fig. 3. The following result is proven in the appendix.

Theorem 2 *Constraint* (32) *corresponding to* $J = J(S_0, S_1, \ldots, S_{H+1})$ (*and* $C = S_H \cup S_{H+1}$) *defines a facet of* $P(n, H)$ *if and only if* $|S_H| = |S_{H+1}| = 1$.

The second example of a new family of inequalities can obtained by combining several jump inequalities in a "circular" fashion. Consider a partition of the node set $V\backslash\{0\}$ into $H+1$ subsets $A_1, A_2, \ldots, A_H, A_{H+1}$ and consider $H+1$ lifted jump inequalities (27) where the subsets are as follows. First inequality: $S_0=\{0\}, S_1=A_1, S_2=A_2, \ldots, S_H=A_H, S_{H+1}=A_{H+1}$; second inequality: $S_0=\{0\}, S_1=A_2, S_2=A_3, \ldots, S_H=A_{H+1}, S_{H+1}=A_1; \ldots;$ Hth inequality:$S_0=\{0\}, S_1=A_H, S_2=A_{H+1}, \ldots, S_H=A_{H-2}, S_{H+1}=A_{H-1}$; and $(H+1)$th inequality $S_0=\{0\}, S_1=A_{H+1}, S_2=A_1, \ldots, S_H=A_{H-1}, S_{H+1}=A_H$. After adding all of these inequalities, using (3) and (4) on the right-hand side and discarding some variables, we obtain the following inequality (for simplicity we omit the derivation here):

$$Hx[\{0\}, V\backslash\{0\}] + \sum_{i=1}^{H+1} \sum_{\substack{j=1; j\neq i,\\ (j-1)\neq i \bmod H}}^{H+1} x[S_i, S_j] \geq H+1. \tag{34}$$

For example, if $n=4$, $H=3$, $A_0=\{0\}, A_1=\{1\}, A_2=\{2\}, A_3=\{3\}, A_4=\{4\}$, we obtain

$$3x_{01}+3x_{02}+3x_{03}+3x_{04}+x_{13}+x_{14}+x_{21}+x_{24}+x_{31}+x_{32}+x_{42}+x_{43} \geq 4. \tag{35}$$

Experiments with small instances have shown that these inequalities are not redundant in the linear programming relaxation of the Hop-MCF formulation, so they contribute to strengthen Hop-MCF or Jump formulations. However, they usually do not define facets of $P(n, H)$.

4 The branch-and-cut algorithm

The branch-and-cut algorithm used to solve model Hop-Cut is based on the algorithm proposed by Poggi de Aragão, Uchoa and Werneck [25] for the STP. The main improvement of this algorithm over previous algorithms based on the same directed cut model, such as the one presented in [19], lies in the effective use of dual and primal heuristics to fix variables and speed up the convergence of the cutting plane generation. The following algorithmic elements were incorporated into the code used to solve model Hop-Cut:

1. *Dual Ascent Heuristic* Initially proposed by Wong [32] for the directed multi-commodity flow model, it can be adapted in a straightforward way for the directed cut model. This fast heuristic usually yields quite good lower bounds, typically less than 5% below the optimal value.
2. *Hot-Starting the Cut Generation* The Dual Ascent Heuristic may be used to provide a good initial set of cuts to be used in the cutting plane generation. In fact, by solving the linear program with those cuts one always obtain a lower bound at least as good as the one provided by Dual Ascent Heuristic. This hot-start may greatly reduce the number of cut rounds required to solve the linear programming relaxation of the model.

3. *Using the Fractional Solutions to Guide Primal Heuristics* An initial primal solution is obtained by applying the well-known Prim Shortest Path Heuristic (Prim-SPH) (see [28]) over the layered graph and improving it by the key-path and node local search [26]. The solutions obtained in this way are only moderately good. However, after each iteration of the cutting plane method, the fractional solutions of the linear programs are used to compute pseudo-costs, that are, then, fed to the Prim-SPH. Those pseudo-costs give incentives for using arcs that appear consistently in those fractional solutions. The subsequent local search is then applied using the original costs.
4. *Fixing by Reduced Costs* The value of a known primal solution permits to remove variables from the model by using reduction tests based on arc reduced costs provided either by the Dual Ascent Heuristic or by subsequent linear programs. The removed variables would never appear in any improving solution. The fixing by reduced costs can be very effective when both primal and dual solutions are close to optimal.

More details about the above mentioned elements can be found in [25,29,31].

5 The diameter constrained spanning tree problem

In this section we adapt the previous methods to a variation of the HMSTP, the DMSTP. Given a graph $G = (V, E)$ with node set $V = \{1, \ldots, n\}$ and edge set E as well as a positive cost c_e associated with each edge e of E, we wish to find a minimal spanning tree with a bound D on the diameter of the tree, which is the maximum number of edges in any of its paths. When $D = 2$ or 3, the problem is easy to solve. However, it is NP-Hard when $D \geq 4$ (see [7]). As noted before, the DMSTP differs from the HMSTP in the sense that here we constrain the path between each pair of nodes while in the HMSTP, only the paths from the special node are constrained. This observation suggests that the DMSTP appears to be much more complex than the HMSTP.

However, several approaches for the DMSTP (see, for instance [1,11,16,27]) have used the properties of tree centers in order to transform the DMSTP into special versions of the HMSTP. For instance, with respect to situations with parameter D even, the following center property proves to be very useful.

Property 1 *A tree T has diameter no more than an even integer D if and only if some node p of T satisfies the property that the path from node p to any other node of the tree contains at most $D/2$ edges.*

This property permits us to transform the DMSTP, for situations when D is even, into an HMSTP in an enlarged graph with an extra root node, node 0, and which is connected to any other node by an edge of zero cost. In the enlarged graph, one must determine a hop constrained spanning tree with depths at most $D/2 + 1$ and such that the degree of the extra node is equal to 1 (the edge emanating from this node will determine the node of the original graph that will serve as the center of the diameter constrained tree). The extra degree constraint is easy to incorporate in the code described in the previous section. Thus, the complexity of solving the DMSTP with diameter D even is essentially the same as solving the HMSTP.

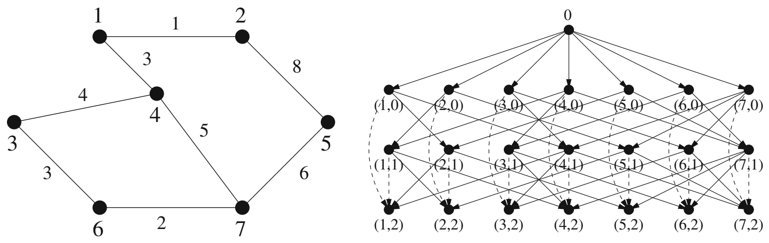

Fig. 6 Transformation of DMSTP for D even: original graph of an instance with $D = 4$ on the *left-hand side* and the corresponding layered graph on the *right-hand side*

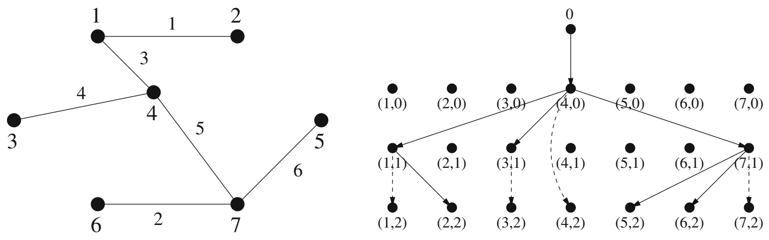

Fig. 7 Optimal DMST in the original graph (cost 21) and its corresponding Steiner tree in G_L ($D = 4$)

In our computations we use the model described in Sect. 2 with the small adaptation mentioned in the previous paragraph. An example depicting the transformation is shown in Figs. 6 and 7. Since the model in [11] is the same as the model in [10], we can use arguments similar to those in Sect. 3 to say that for D even, the linear programming relaxation of the adapted layered graph model we use here is at least as good as the linear programming relaxation of the model proposed by Gouveia and Magnanti [11].

For situations when D is odd, there appears not to be any agreement in what should be the best center property to use. The reason for this is that the straightforward adaptation of the center property for D odd ("a tree T has diameter no more than D if and only if some edge (p, q) of T satisfies the property that the path from any other node of the tree to p or q contains at most $(D - 1)/2$ edges") is not easy to combine either with a Network Flow model (as in [10]) or with the layered graph approach proposed here unless one allows the creation of special nodes corresponding to the edges of the graph (as suggested and tested in [11]). However, this modification leads to oversized networks and the proposed approaches on these networks usually take much more time (when computer storage requirements do not become a bigger problem) than similar approaches used for situations with D even. Alternative approaches have been suggested in [11,13] where 2-path (or 2-flow) and 2-tree based models were efficiently combined with 2-center properties for trees with diameter odd. However, these properties do not appear to be easily combined with the layered graph approach proposed in the previous sections.

A different and perhaps less straightforward transformation for the situations when D is odd and which is suitable to be modeled by the layered graph approach is described as follows: add two special dummy layers (0 and -1) corresponding to the possible

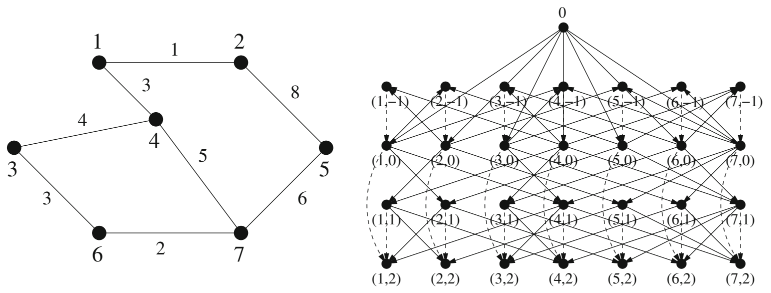

Fig. 8 Transformation of DMSTP for D odd: original graph of an instance with $D = 5$ on the *left-hand side* and the corresponding layered graph on the *right-hand side*

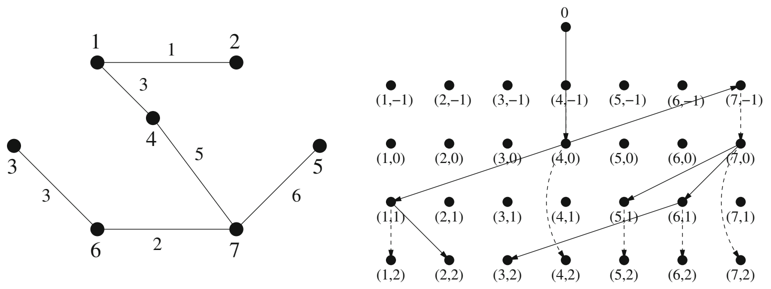

Fig. 9 Optimal DMST in the original graph (cost 20) and its corresponding Steiner tree in G_L ($D = 5$)

Table 1 Remaining size of reduced HMSTP instances

n	20 (%)	40 (%)	60 (%)	80 (%)	100 (%)	120 (%)	160 (%)
TC	26	27	25	25	25	27	53
TE	70	67	70	68	70	75	78
TR	44	46	51	46			

choices of nodes to be the extremes of the central edge of the tree. Node 0 is connected to all nodes in layer 0 with zero cost arcs. We add a degree constraint guaranteeing that exactly one of these edges is chosen. Every node in layer 0, $(i, 0)$, is linked to any node $(j, -1)$, such that $(i, j) \in E$, in layer -1 by an arc of cost c_{ij}. We also add a constraint stating that exactly one of these edges is chosen. This constraint and the previously described constraint guarantee that the two extremes of the central edge are chosen and its cost is accounted. Then we add arcs of cost 0 linking each pair of nodes $(i, -1)$ and $(i, 0)$. In this way we guarantee that the two nodes $(i, 0)$ and $(j, 0)$ are selected if edge (i, j) is the central edge of the tree. The remainder of the layers of the graph are as before. Figure 8 shows a graph and the corresponding layered graph for $D = 5$ and Fig. 9 gives an example of a corresponding solution in the two graphs.

Table 2 TC instances: results of the branch-and-cut method over the Hop-Cut model

n	H	OPT	LP	DA	PRIM-HOP	NCUTS	*T* (s)
	3	340	340	340	340	–	0
20	4	318	318	318	318	–	0
	5	312	312	312	312	–	0
	3	609	609	601	609	190	0.06
40	4	548	548	540	548	213	0.07
	5	522	522	516	524	337	0.24
	3	866	866	857	892	479	0.35
60	4	781	781	775	795	834	1.26
	5	734	734	732	734	153	0.15
	3	1,072	1,072	1,066	1,084	556	0.49
80	4	981	981	973	995	2,028	46.9
	5	922	922	920	934	1,286	6.57
	3	1,259	1,259	1,237	1,290	1,843	15.3
100	4	1,166	1,166	1,158	1,198	3,206	190
	5	1,104	1,104	1,098	1,116	3,590	251
	3	1,059	1,059	1,051	1,102	1,186	2.68
120	4	926	926	921	962	1,743	25.9
	5	853	853	849	875	2,369	103
	3	1,357	1,357	1,349	1,426	2,683	160
160	4	1,133	1,133	1,130	1,163	3,495	643
	5	1,039	1,039	1,033	1,055	8,431	10,292

Table 3 TR instances: results of the branch-and-cut method over the Hop-Cut model

n	H	OPT	LP	DA	PRIM-HOP	NCUTS	*T* (s)
	3	168	168	168	168	–	0
20	4	146	146	146	146	–	0
	5	137	137	137	137	–	0
	3	176	176	172	177	126	0.05
40	4	149	149	147	149	154	0.13
	5	139	139	139	139	–	0
	3	213	213	206	217	316	0.17
60	4	152	152	151	153	240	0.18
	5	124	124	124	124	–	0.02
	3	208	208	206	208	170	0.14
80	4	180	180	178	180	267	0.36
	5	164	164	164	164	–	0.07

Table 4 TE instances: results of the branch-and-cut method over the Hop-Cut model

n	H	OPT	LP	DA	PRIM-HOP	NCUTS	*T* (s)
	3	449	449	449	457	114	0.03
20	4	385	385	385	385	–	0.01
	5	366	366	361	366	122	0.05
	3	708	708	708	728	301	0.13
40	4	627	627	624	629	267	0.14
	5	590	590	589	596	398	0.41
	3	1,525	1,525	1,521	1,569	488	0.48
60	4	1,336	1,336	1,328	1,373	923	3.43
	5	1,225	1,225	1,225	1,229	374	0.46
	3	1,806	1,806	1,802	1,840	753	1.24
80	4	1,558	1,558	1,549	1,580	917	4.29
	5	1,442	1,442	1,435	1,477	2,878	226
	3	2,092	2,092	2,082	2,111	1,344	9.85
100	4	1,788	1,788	1,771	1,888	2,887	356
	5	1,625	1,625	1,625	1,699	1,592	36.6
	3	1,267	1,267	1,258	1,352	1,708	15.6
120	4	1,074	1,074	1,071	1,155	3,043	574
	5	969	969	962	1,000	5,071	3,397
	3	1,496	1,496	1,488	1,616	2,282	76.2
160	4	1,229	1,229	1,221	1,286	6,458	5,626
	5	1,107	1,106.5	1,098	1,182	15,764	53,797

6 Computational results

In this section we give results with the branch-and-cut method described in Sect. 4 for the two problems, the HMSTP and the DMSTP. The tests were performed on a PC Intel Core 2 Duo, 2.2 GHz, with 2 GB of RAM. The method was implemented using the callable routines in XPRESS-MP 2007A, that both solves linear programs and may perform branch to obtain the optimal integer solutions.

6.1 Computational results for the HMSTP

The experiments use some complete graph instances on 21, 41, 61, 81, 101, 121 and 161 nodes that have been used in previous papers for the HMSTP (see, for instance [6]) and for other constrained spanning trees, namely the capacitated minimum spanning tree problem (see, for instance [30]). For each size, we have considered one random cost instance, denoted by TR, and two Euclidean cost instances. Two locations for the root are considered for the Euclidean instances: Instances that have the root located in the center of the grid, denoted by TC, and instances that have the root located on a

Table 5 Comparing the Hop-Cut and Hop-MCF models

PROB, *n*	H	OPT	LP_HOP-CUT	*T* (s)	LP_HOP-MCF	*T* (s)
	3	609	609	0.06	604.5	$2 + 48$
TC, 40	4	548	548	0.07	547	$8 + 3$
	5	522	522	0.24	522	13
	3	176	176	0.05	176	2
TR, 40	4	149	149	0.13	148.3	$9 + 3$
	5	139	139	0	139	$26 + 1$
	3	708	708	0.13	701.7	$175 + 1{,}984$
TE, 40	4	627	627	0.14	625.3	$969 + 9{,}797$
	5	590	590	0.41	588.1	$2{,}794 + 23{,}052$
	3	1,072	1,072	0.49	1,069	$157 + 3{,}204$
TC, 80	4	981	981	46.9	976.5	$1{,}811 + 23{,}659$
	5	922	922	6.57	920.6	$4{,}198 + 7{,}590$
	3	208	208	0.14	208	$64 + 3$
TR, 80	4	180	180	0.36	180	$676 + 4$
	5	164	164	0.07	164	$1{,}271 + 6$
	3	1,806	1,806	1.24	1,792.5	$16{,}127 + (r)$
TE, 80	4	1,558	1,558	4.29	1,544.5	$160{,}127 + (r)$
	5	1,442	1,442	226	–	–

Values in the last two columns are taken from [6], their time values are split into root time + remaining time to optimal, (r) means that only the root node was executed

corner of the grid, denoted by TE. The hop parameter H was set to 3, 4 and 5 in every case.

In order to reduce the size of each instance, we have used the following simple arc elimination test (see [9]). If $c_{ij} > c_{0j}$, then any optimal solution does not uses arc (i, j) and if $c_{ij} = c_{0j}$ $(i \neq 0)$, then there is an optimal solution without arc (i, j). This means that arc (i, j) can be eliminated whenever $c_{ij} \geq c_{0j}$. This arc elimination test is applied to every instance before its solution. Table 1 shows, for each instance, the percentage of number of arcs still remaining after the elimination test was performed.

Tables 2, 3 and 4 give information on the performance of the method for the TC, TR and TE instances. The first column gives the number of nodes (without the root node) of the corresponding graph. The second column gives the value of H. The third column gives the value of the integer optimal value and the fourth gives the value of the bound produced by the Hop-Cut model. The next columns give, respectively, the value of the lower bound obtained by Dual Ascent Heuristic, the value of the upper bound obtained by the first application of Prim-SPH + local search and the number of inequalities (2) added in the branch-and-cut algorithm. Finally, the last column gives the total CPU time in seconds spent by the whole method (a time of 0 indicates that the instance was solved in less than 1 ms, usually because both Dual Ascent and Prim-SPH found the same bound). In all instances, except on instance TE160 with $H = 5$, the

Table 6 Comparing DMSTP methods: instances from [27]

$\|V\|$	$\|E\|$	D	OPT	Hopcut		Lift DMST		ILP		GMR04/GM03		CP
				Gap	T (s)	Gap	T (s)	Gap	T (s)	Gap	T (s)	T (s)
		4	346	0	0.05	23.9	4.7	22	0.7	0.9	3.09	0.08
		5	331	0	0.06	28.9	22.8	29.2	3	0.5	5.7	0.22
15	105	6	314	0	0.06	16.4	18.6	32.8	8.1	0.5	163.69	0.28
		7	303	0	0.07	10.6	26.9	10.6	20	0.0	6.23	0.38
		9	290	0	0.07	6.6	6.2	6.6	10.7	0.0	7.03	0.47
		10	286	0	0.08	8.3	1	5.3	4.3	0.0	10.59	0.41
		4	349	0	0.07	24.4	562.9	25	2.5	0.1	11.97	0.2
		5	414	0	0.08	20.9	436.7	17.8	8.1	1.1	173.59	1.06
20	190	6	298	0	0.06	12.5	455.2	21.7	95	2.2	1,590.03	2.03
		7	333	0	0.01	7.7	5.1	5	4.5	0.1	13.17	0.97
		9	327	0	0.01	8.9	73.4	8.8	66.7	0.0	27.31	5.01
		10	324	0	0.01	8.1	29.7	8.1	101	0.0	41.25	6.08
		4	500	0	0	26.6	15,203.7	23.2	12	0.0	55.59	1.48
		5	429	0	0.07	22.6	>20,000	23.1	64.3	0.3	55.81	2.83
25	300	6	378	0	0.11	11.4	826.7	17	26.4	0.0	193.67	39.14
		7	408	0	0.13	15.9	11,521.3	17.6	770.5	1.1	2,647.51	56.06
		9	336	0	0.11	8	246	5.2	295.4	0.0	103.27	114.14
		10	379	0	0.11	5.9	254.8	5.3	404.9	0.0	210.2	55.47
		4	442	0	0.03	16.5	1	13.4	0.2	0.2	0.56	0.05
		5	381	0	0	57.8	4.6	58.8	1	0.0	0.52	0.17
20	50	6	329	0	0	9.5	0.8	11.6	5.1	0.0	0.91	0.13
		7	362	0	0.06	8.4	0.8	25.5	1.2	0.0	0.75	0.14
		9	362	0	0.01	7.7	0.7	10.4	2.8	0.0	0.59	0.45
		10	359	0	0	3.9	0.2	2.2	1.3	0.0	2.28	0.64
		4	755	0	0.07	27.7	43.7	19.4	1.9	0.0	1.45	5.44
		5	729	0	0.08	52.9	291.7	47.5	6.4	0.0	2.39	7.31
40	100	6	599	0	0	4.6	50.9	6.4	13.2	0.0	6.03	4.72
		7	667	0	0.09	14.7	459.4	34.5	212.4	0.2	6.77	34.38
		9	552	0	0.13	15.8	1,565	21.1	979.8	0.0	10.81	40.16
Averages:				0	0.06	16.80	1,146.95	18.45	107.70	0.25	184.58	13.10

solution of the linear programming relaxation of the Hop-Cut model was integral and no branching was performed.

Table 5 compares the new method, in terms of bounds and CPU times (see column LP-HOP-CUT and subsequent column), with the lower bounds taken from the model Hop-MCF described in Sect. 3 (see column LP-G and subsequent column), on instances with $n = 40$ and $n = 80$. These results show that the theoretically dominance of the new lower bound is reflected in practice. They also show that the new approach (namely, the branch-and-cut method applied to the theoretical strong

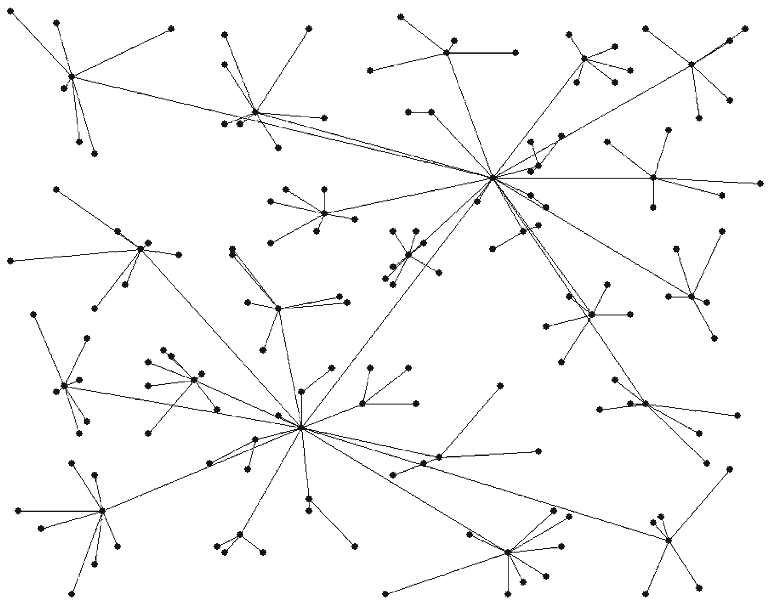

Fig. 10 Optimal solution of DMSTP instance te161 with $D = 5$

Hop-Cut model) solves quite easily instances with up to 160 nodes. Previously, only some instances with up to 80 nodes have been solved. The computational results of the model Hop-MCF were obtained on a PC Pentium IV, 2.4 GHz with 768 MB of RAM and used the callable routine CPLEX 7.1.

6.2 Computational results for the DMSTP

For the computational experiments concerning the DMSTP we have performed two groups of tests. In the first group, our aim is to compare the method described in this paper with the other methods described in the recent literature. In the second group we consider much larger instances in order to show that the method proposed in this paper can solve many instances that can not be solved by previous methods.

For the first group we report results given by the methods described in [11,16,27] (for situations with D even), [12] (for situations with D odd), and [23]. The first method uses formulations based on enhanced versions of the Miller–Tucker–Zemlin constraints. The second uses formulations based on precedence variables. The method described in [11] is based on the Hop-MCF formulation readapted for the DMSTP as explained in Sect. 6.2. The method described in [12] is based on two center properties of D diameter situations. The more recent method [23] is based on constraint programming. These instances are taken from [27] and include dense graph instances with up to 40 nodes. Although the methods described in [11,12] have been originally tested for sparse graph instances, we have specially run them (on a Pentium IV 2.4 GHz) for these dense graph instances in order to permit us to compare all methods in the same

Table 7 Results of the Hop-Cut DMSTP model over TC instances

D	$\|V\|$	OPT	LP	T (s)	Nodes	$\|V\|$	OPT	LP	T (s)	Nodes
4		747	747	0.27	1		1,083	1,081.5	1.32	5
5		673	673	0.38	1		977	977	3.98	1
6		606	606	4.49	1		856	856	6.86	1
7		575	575	2.14	1		821	821	5.96	1
8	41	544	544	6.05	1	61	781	781	386	1
9		532	532	2.22	3		760	760	9.87	1
10		516	516	9.73	1		734	734	91.8	1
11		508	508	0.72	1		722	722	5.91	1
12		498	498	6.55	1		712	712	7.38	1
4		1,303	1,303	1.39	1		1,549	1,549	9.58	1
5		1,203	1,203	8.36	1		1,438	1,438	362	1
6		1,064	1,064	226	1		1,259	1,259	1,858	1
7		1,024	1,024	169	1		1,220	1,220	1,333	1
8	81	976	976	2,933	1	101	1,158	1,158	4,574	1
9		952	952	1,869	1		1,136	1,136	522	1
10		920	920	2,341	1		1,104	1,104	31,718	1
11		906	906	2,510	1		1,088	1,088	42,182	1
12		884	884	776	1		1,060	1,060	7,416	1
4		1,431	1,431	6.82	1		1,731	1,731	19.7	1
5		1,281	1,281	42.7	1		1,572	1,572	8,377	1
6		1,059	1,059	1,183	1		1,247	1,245.83	13,611	11
7		1,005	1,005	322	1		1,177	1,177	15,560	1
8	121	925	925	4,023	1	161	–	–	–	–
9		894	894	5,636	1		–	–	–	–
10		851	851	13,901	1		–	–	–	–
11		836	836	36,925	1		–	–	–	–
12		812	812	29,596	1		–	–	–	–

data set. Table 6 shows for each method, the percent root gap (except for the constraint programming method) and the total time in seconds to solve the instance. Times from [16,27] were both obtained on a Pentium IV 2.8 GHz, those from [23] on a Pentium IV 3 GHz. The last row in the table shows averages.

For the second group we use the same instances as the ones used for the HMSTP, with up to 161 nodes. In these results (see Tables 7, 8, 9) one can, again, see that the proposed method is quite strong in the sense that for almost all tested instances the linear programming bound is optimal. It is also interesting to see the effect of the new transformation for situations when D is odd. In our results, it does not appear to be any significant difference between the two cases, D even and D odd, which is in contrast with previous transformations/approaches. Figure 10 depicts an optimal solution of the TE instance with 161 vertices, for $D = 5$.

Table 8 Results of the Hop-Cut DMSTP model over TE instances

D	|V|	OPT	LP	T (s)	Nodes	$\|V\|$	OPT	LP	T (s)	Nodes
4		742	742	0.31	1		1,657	1,657	0.68	1
5		678	678	0.76	1		1,528	1,528	2.72	1
6		606	606	0.43	1		1,317	1,317	25.1	2
7		585	585	1.19	1		1,255	1,255	15.9	1
8	41	562	562	2.54	1	61	1,174	1,174	6.16	1
9		553	552.5	53.1	1		1,139	1,139	44.1	1
10		537	537	0.15	1		1,083	1,083	1.04	1
11		529	529	1.52	1		1,063	1,063	2.89	1
12		525	525	30.7	1		1,044	1,044	21.1	1
4		2,045	2,045	2.11	1		2,377	2,377	4.07	1
5		1,828	1,828	6.39	1		2,166	2,166	217	1
6		1,564	1,564	12.8	1		1,802	1,802	257	1
7		1,479	1,479	39.8	1		1,708	1,708	352	1
8	81	1,399	1,399	963	1	101	1,585	1,585	1,037	2
9		1,355	1,355	3,786	1		1,545	1,545	428	1
10		1,293	1,293	20.4	1		1,486	1,486	879	1
11		1,267	1,267	30.1	1		1,453	1,453	1,302	1
12		1,232	1,232	309	1		1,415	1,415	5,540	1
4		1,448	1,448	6.72	1		1,741	1,741	23.8	1
5		1,297	1,297	4.31	1		1,583	1,583	6,918	1
6		1,077	1,077	2,120	1		1,253	1,251.83	14,422	11
7		1,019	1,019	333	1		1,182	1,182	20,812	1
8	121	938	938	4,213	1	161	1,081	1,081	158,572	1
9		907	907	415	1		–	–	–	–
10		866	866	9,565	1		–	–	–	–
11		848	848	4,859	1		–	–	–	–
12		826	826	74,178	2		–	–	–	–

7 Conclusion

This paper presented an effective method for the HMSTP and for the DMSTP, basically reducing them to a classical STP in a layered graph. Moreover, on the theoretical side, we provided a number of results explaining why the new extended HMSTP formulation is significantly stronger than previous formulations. In particular, we have shown that combinations of very simple inequalities in the extended space project into more complex and previously unknown inequalities (including proven facets) in the original space.

The methods provided in this paper can be easily adapted for a number of similar problems. For example, there can be a HMSTP variant where the hop constraint $H(i)$ depends on the node i. This variant can be useful in applications where more

Table 9 Results of the Hop-Cut DMSTP model over TR instances

D	$\|V\|$	OPT	LP	T (s)	Nodes	$\|V\|$	OPT	LP	T (s)	Nodes
4		249	249	0.05	1		264	264	0.15	1
5		198	198	0.43	1		213	211.25	3.34	5
6		155	155	0.24	1		155	155	7.47	1
7		144	144	0.75	1		140	140	25.1	1
8	41	135	135	0.9	1	61	124	124	7.12	1
9		131	131	0.48	1		117	117	8.75	1
10		129	129	1.04	1		114	114	8.32	1
11		128	128	2.08	1		112	111.5	13.7	1
12		127	127	0.22	1		108	108	0.57	1
4		416	416	0.93	1					
5		326	324	884	15					
6		208	208	85.8	1					
7		187	187	91.3	1					
8	81	168	167.25	8.41	1					
9		160	160	14.7	1					
10		153	153	2.39	1					
11		152	152	2.41	1					
12		151	151	1.65	1					

important nodes need to be closer to the central node. Another potential application is the Hop-Constrained and DMSTP, where only a subset of the nodes need to be connected by a tree respecting the hop or diameter constraint. Those two problems arise frequently in multicast routing with quality of service requirements [24].

Acknowledgments LG is supported by FCT, grant POCTI-ISFL-1-152; LS is supported by CNPq, grant 140489/2004-5; EU received support from CNPq grant 304533/02-5 and FAPERJ grant E-26/100.504/2007. We also thank two anonymous referees for carefully reading the submitted manuscript.

Appendix A: Proof of Theorem 2

Proof Remember that we are assuming that $H \geq 3$ and $n \geq H + 1$. First, suppose that S_H and S_{H+1} are singletons. For an arc set $T \subseteq A$, define $\chi(T)$ as the incidence vector of T in the R^{n^2} space. Let $EQ(J)$ denote the set of points of $P(n, H)$ that satisfy the jump-cut with equality. Suppose that $\alpha^T x = b$ defines a hyperplane that contains $EQ(J)$. Normalizing the coefficients using the indegree equalities (8) for each node $v \in V \backslash \{0\}$, we can assume that $\alpha_{0v} = 0$ if $v \in S_1$, $\alpha_{0v} = 1$ if $v \in S_i$, $i < H$, and that $\alpha_{0v} = 2$ otherwise. For each j, $1 \leq j \leq H + 1$, and $s \in S_j$, we define the arc sets $A(s) = \{(s, l) : l \in S_{j+1}\}$, $B(s) = \{(s, l) : l \in S_j, l \neq s\}$ and $C(s) = \{(s, l) : l \in S_{j-1}\}$. Pick nodes $s_k \in S_k$ for $k = 0, \ldots, H + 1$.

- Suppose that $e \in \{(s_i, s_j) : 0 < i < j-1, j < H\}$. Define the arc set $T^1(j)$ as $\cup_{k \neq j-1} A(s_k) \cup B(s_j) \cup \{(0, s_j)\}$. It can be seen that $\chi(T^1(j)) \in EQ(J)$. The same holds for $\chi(T^1(j) \backslash \{(0, s_j)\} \cup \{e\})$. So $\alpha_e = \alpha_{0s_j} = 1$. This argument does not depend on the particular choice of a node s_k for each set S_k. Therefore, $\alpha_e = 1$ for any arc e from S_i to S_j, $0 < i < j-1$, $j < H$. Similar reasonings are implicit in the remaining of the proof.
- Suppose that $e \in \{(s_i, s_j) : 0 < i < H-1, j \geq H\}$. Observe that $\chi(T^1(j) \backslash (0, s_j) \cup \{e\}) \in EQ(J)$. Therefore $\alpha_e = \alpha_{0s_j} = 2$.
- Suppose that $e \in \{(s_i, s_1) : 1 < i \leq H\} \cup \{(s'_1, s_1) : s'_1 \in S_1 \backslash \{s_1\}\}$. Define the arc set $T^2(j)$ as $\cup_{k \geq j} A(s_k) \cup_{k \leq j-3} A(s_k) \cup B(s_j) \cup C(s_j) \cup \{(0, s_j)\}$, $\chi(T^2(j)) \in EQ(J)$, $j \geq 2$. The vector $\chi(T^2(3) \backslash \{(0, s_1)\} \cup \{e\})$ also belongs to $EQ(J)$. Therefore $\alpha_e = \alpha_{0s_1} = 0$.
- Note that the symmetric difference $T^1(H) \ominus T^2(H+1)$ is $\{(s_H, s_{H+1}), (s_{H+1}, S_H)\}$, therefore $\alpha_{s_{H+1}s_H} = \alpha_{s_H s_{H+1}} = \beta$.
- Suppose that $e \in \{(s_i, s_j) : 2 \leq j \leq H-1, i > j\} \cup \{(s'_j, s_j) : s'_j \in S_j \backslash \{s_j\}, 2 \leq j \leq H-1\}$. Observe that both $\chi(T^1(j+1))$ and $\chi(T^1(j+1) \backslash \{(s_{j-1}, s_j)\} \cup \{e\})$ belong to $EQ(J)$. Therefore $\alpha_e = \alpha_{s_{j-1}s_j} = \gamma_{s_j}$. Suppose that $s_i, s_j \in \{s_k : 2 \leq k \leq H-1\}$ and $s_i \neq s_j$. Note that $T^1(j) \backslash T^1(i) = \{(0, s_j)\} \cup A(s_{i-1}) \cup B(s_j)$, that $T^1(i) \backslash T^1(j) = \{(0, s_i)\} \cup A(s_{j-1}) \cup B(s_i)$, and $\alpha_{0s_i} = \alpha_{0s_j} = 1$. We prove that $\alpha_{s_{i-1}v} = \alpha_{s_i v} = \gamma_v$, $(s_{i-1}, v) \in A(s_{i-1}) \backslash \{(s_{i-1}, s_i)\}$ and $(s_i, v) \in B(s_i)$. The same holds for $A(s_{j-1})$ and $B(s_j)$. Therefore $\gamma_{s_i} = \gamma_{s_j} = \gamma$.
- Define T^3 as $\cup_{k=0}^{H} A(s_k) \backslash \{(s_{H-2}, s_{H-1})\} \cup \{(s_1, s_{H-1})\}$, $\chi(T^3) \in EQ(J)$. The same holds for $\chi(T^3 \backslash \{(s_{H-1}, s_H)\} \cup \{(s_{H-1}, s_{H+1})\})$. Therefore $\alpha_{s_{H-1}s_H} = \alpha_{s_{H-1}s_{H+1}}$. Using $T^1(H)$ and T^3, the coefficients $\alpha_{s_{H-1}s_H}$ can be seen to be equal to $1 + \gamma$.
- Define T^4 as $\cup_{k=0}^{H-1} A(s_k) \cup \{(s_H, s_{H+1})\}$. Both $\chi(T^4)$ and $\chi(T^1(H))$ belong to $EQ(J)$. Therefore $\beta = 2\gamma$.

Summarizing, $\alpha^T x = b$ must have the following format:

$$\sum_{(p,q) \in J} x_{pq} + \sum_{(p,q) \in C} x_{pq} - x\left[S_{H-1}, S_{H+1}\right]$$
$$+\gamma \left[\sum_{k=2}^{H-1} x[V \backslash \{0\}, S_K] + \sum_{i \in S_{H-1}} (x_{is_H} + x_{is_{H+1}}) + 2(x_{s_H s_{H+1}} + x_{s_{H+1} s_H}) \right] = b. \tag{36}$$

By adding to $\alpha^T x = b$ the following combination of constraints (8):

$$-\gamma \left(\sum_{j \in V \backslash \{0, S_1, S_H, S_{H+1}\}} \sum_{(i,j) \in \delta^-(j)} x_{ij} \right) - 2\gamma \left(\sum_{j \in S_H \cup S_{H+1}} \sum_{(i,j) \in \delta^-(j)} x_{ij} \right)$$
$$= -\gamma |V \backslash \{0, S_1, S_H, S_{H+1}\}| - 4\gamma,$$

one obtains $(1-\gamma)(\sum_{(p,q)\in J} x_{pq} + \sum_{(p,q)\in C} x_{pq} - x[S_{H-1}, S_{H+1}]) = b - \gamma (|V\backslash\{0, S_1, S_H, S_{H+1}\}| + 4) = b'$. Substituting any solution from $EQ(J)$ in this equation, we have that $b' = 2(1-\gamma)$. Therefore $\alpha^T x = b$ is equivalent either to the trivial equation $0^T x = 0$ or to a scalar multiple of the jump-cut inequality.

Now suppose that $|S_{H+1}| > 1$ and let $u \in S_{H+1}$. Then the jump-cut constraint associated with $J(S_1, \ldots, S_H \cup \{u\}, S_{H+1}\backslash\{u\})$ dominates the jump-cut constraint associated with $J(S_1, \ldots, S_{H+1})$. Finally, suppose that S_{H+1} is a singleton but $|S_H| > 1$. In this case, all points in $EQ(J)$ satisfy the following equality:

$$\sum_{k=1}^{H-1} x[S_0, S_k] = x[S_{H-1}S_H] + x[S_{H-1}S_{H+1}]. \tag{37}$$

Remark that if S_H is a singleton, T_4 is an arc set that does not satisfy (37). On the other hand, if $|S_H| > 1$, T_4 can not be extended to an arc set whose incidence vector belongs to $EQ(J)$. For example, $T_4 \cup B(S_H)$ does not satisfy the hop limit. □

References

1. Achuthan, N.R., Caccetta, L., Caccetta, P.A., Geelen, J.F.: Computational methods for the diameter restricted minimum weight spanning tree problem. Australas. J. Comb. **10**, 51–71 (1994)
2. Chopra, S., Rao, M.R.: The Steiner tree problem I: formulations, compositions and extension of facets. Math. Program. **64**(2), 209–229 (1994)
3. Dahl, G.: The 2-hop spanning tree problem. Oper. Res. Lett. **23**, 21–26 (1998)
4. Dahl, G.: Notes on polyhedra associated with hop-constrained paths. Oper. Res. Lett. **25**, 97–100 (1999)
5. Dahl, G., Flatberg, T., Foldnes, N., Gouveia, L.: The jump formulation for the hop-constrained minimum spanning tree problem. Tech. Rep. CIO 5, University of Lisbon (2004)
6. Dahl, G., Gouveia, L., Requejo, C.: On formulations and methods for the hop-constrained minimum spanning tree problem. In: Pardalos, P., Resende, M. (eds.) Handbook of Optimization in Telecommunications, pp. 493–515. Springer, Berlin (2006)
7. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, New York (1979)
8. Gouveia, L.: Using the Miller–Tucker–Zemlin constraints to formulate a minimal spanning tree problem with hop constraints. Comput. Oper. Res. **22**(9), 959–970 (1995)
9. Gouveia, L.: Multicommodity flow models for spanning trees with hop constraints. Eur. J. Oper. Res. **95**(1), 178–190 (1996)
10. Gouveia, L.: Using variable redefinition for computing lower bounds for minimum spanning and Steiner trees with hop constraints. INFORMS J. Comput. **10**(2), 180–188 (1998)
11. Gouveia, L., Magnanti, T.L.: Network flow models for designing diameter-constrained minimum-spanning and Steiner trees. Networks **41**(3), 159–173 (2003)
12. Gouveia, L., Magnanti, T.L., Requejo, C.: A 2-path approach for odd-diameter-constrained minimum spanning and Steiner trees. Networks **44**(4), 254–265 (2004)
13. Gouveia, L., Magnanti, T.L., Requejo, C.: An intersecting tree model for odd-diameter-constrained minimum spanning and Steiner trees. Ann. Oper. Res. **146**, 19–39 (2006)
14. Gouveia, L., Requejo, C.: A new Lagrangian relaxation approach for the hop-constrained minimum spanning tree problem. Eur. J. Oper. Res. **132**(3), 539–552 (2001)
15. Gouveia, L., Simonetti, L., Uchoa, E.: Modelling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs. Tech. Rep. RPEP, vol. 8, no.7, Universidade Federal Fluminense, Engenharia de Produção, Niterói, Brazil (2008)
16. Gruber, M., Raidl, G.R.: A new 0-1 ILP approach for the bounded diameter minimum spanning tree problem. In: Proceedings of the 2nd International Network Optimization Conference, Lisbon, pp. 178–185 (2005)

17. Hwang, F., Richards, D., Winter, P.: The Steiner Tree Problems. Annals of Discrete Mathematics. North-Holland, Amsterdam (1992)
18. Kerivin, H., Mahjoub, A.R.: Design of survivable networks: a survey. Networks **46**(1), 1–21 (2005)
19. Koch, T., Martin, A.: Solving Steiner tree problems in graphs to optimality. Networks **33**, 207–232 (1998)
20. Maculan, N.: The Steiner problem in graphs. Ann. Discret. Math. **31**, 185–212 (1987)
21. Magnanti, T.L., Wolsey, L.A. : Optimal trees. In: Ball, M. , Magnanti, T.L. , Monma, C., Nemhauser, G. (eds.) Network Models, Handbook in Operations Research and Management Science, vol. 7, pp. 503–615. Elsevier, Amsterdam (1995)
22. Manyem, P., Stallmann, M.F.M.: Some approximation results in multicasting. Tech. Rep. TR-96-03, North Carolina State University (1996)
23. Noronha, T.F., Santos, A.C., Ribeiro, C.C.: Constraint programming for the diameter constrained minimum spanning tree problem. In: Proceedings of IV Latin-American Algorithms, Graphs and Optimization Symposium, Electronic Notes in Discrete Mathematics, vol. 30, pp. 93–98. Puerto Varas, Chile (2008)
24. Oliveira, C., Pardalos, P., Resende, M.: Optimization problems in multicast tree construction. In: Handbook of Optimization in Telecommunications, pp. 701–731. Springer, Berlin (2006)
25. Poggide Aragão, M., Uchoa, E., Werneck, R.: Dual heuristics on the exact solution of large Steiner problems. Electron. Notes Discret. Math. **7**, 150–153 (2001)
26. Ribeiro, C., Uchoa, E., Werneck, R.: A hybrid GRASP with perturbations for the Steiner problem in graphs. INFORMS J. Comput. **14**, 228–246 (2002)
27. Santos, A.C., Lucena, A., Ribeiro, C.C.: Solving diameter constrained minimum spanning tree problems in dense graphs. In: Experimental and Efficient Algorithms, Lecture Notes in Computer Science, vol. 3059/2004, pp. 458–467. Springer, Berlin (2004)
28. Takahashi, H., Matsuyama, A.: An approximate solution for the Steiner problem in graphs. Mathematica Japonica **24**, 573–577 (1980)
29. Uchoa, E.: Algoritmos para problemas de Steiner com aplicações em projeto de circuitos VLSI. Ph.D. thesis, Pontifícia Universidade Católica do Rio de Janeiro (2001)
30. Uchoa, E., Fukasawa, R., Lysgaard, J., Pessoa, A., Poggide Aragão, M., Andrade, D.: Robust branch-cut-and-price for the capacitated minimum spanning tree problem over a large extended formulation. Math. Program. **112**(2), 443–472 (2008)
31. Werneck, R.: Problema de Steiner em grafos: algoritmos primais, duais e exatos. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro (2001)
32. Wong, R.: A dual ascent approach for Steiner tree problems on a directed graph. Math. Program. **28**(3), 271–287 (1984)
33. Woolston, K.A., Albin, S.L.: The design of centralized networks with reliability and availability constraints. Comput. Oper. Res. **15**(3), 207–217 (1988)