Modelling the hop-constrained minimum spanning tree problem over a layered graph

Luis Gouveia, *Universidade de Lisboa, Portugal*
Luidi Simonetti, *Universidade Federal do Rio de Janeiro, Brazil*
Eduardo Uchoa, *Universidade Federal Fluminense, Brazil*

Keywords: Network Design, Integer Programming, Steiner Problem

1. Introduction

The Hop-constrained Minimum Spanning Tree Problem (HMST) 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 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 HMST is NP-hard because it contains as a particular case (the case with H = 2) a NP-Hard version of the Simple Uncapacitated Facility Location problem (see Gouveia [5] and Dahl [1]). Manyem and Stallmann [11] have shown that the HMST is not in APX, i.e., the class of problems for which it is possible to have polynomial time heuristics with a guaranteed approximation bound. The HMST models the design of centralized telecommunication networks with quality of service constraints. The root node represents the site of a central processor (computer) and the remaining nodes represent terminals that are required to be linked to the central processor. The hop constraints limit the number of hops (arcs) between the root node and any other node and guarantee a certain level of service with respect to some performance constraints such as availability and reliability (see Woolston and Albin [16]). Availability is the probability that all the transmission lines in the path from the root node to the terminal are working. Reliability is the probability that a session will not be interrupted by a link failure. In general, these probabilities decrease with the number of links in the path implying that paths with fewer hops have a better performance with respect to availability and reliability. 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 queuing delays and fewer hops mean, in general, lower delays. Lower bounding schemes for the HMST based on network-flow models have been suggested in Gouveia [6, 7] and Gouveia and Requejo [8]. More recently, Dahl et al. [2] propose a formulation involving only natural design variables and exponential sized set of constraints and propose a lower bound based on an appropriate Lagrangean relaxation. The recent paper by Dahl, Gouveia and Requejo [3] summarizes these approaches. The models described in these papers view the HMST problem as defined in the original graph (although some of these models view the underlying hop-constrained shortest path problem as defined in an appropriate layered graph).

In this paper we propose a modelling approach for the HMST that views the whole problem as defined in an appropriate layered directed graph. In fact, this is equivalent to a transformation to a suitable directed Steiner tree problem over that layered graph. Section 2 defines the layered graph and presents the resulting formulation. Section 3 describes the cutting plane algorithm used to solve that formulation. Finally, Section 4 reports computational experiments over a large set of benchmark instances. The overall approach is shown to be very efficient. Instances with up to 160 nodes can be solved in reasonable times, a significant increase with respect to previous methods.

2. Layered Graph Definition and Transformation to Steiner

Consider the layered graph $G_E = (V_E, A_E)$, where

- $V_E = \{0\} \cup \{(i,h) : 1 \leq h \leq H,\ i \in V \setminus \{0\}\}$
- $A_E = \{A_0 = \{(0,(j,1)) : (0,j) \in E\}$
  $\cup\ A_1 = \{((i,h),(j,h+1)) : (i,j) \in E,\ i \neq 0,\ 1 \leq h \leq H-1\}$
  $\cup\ A_2 = \{((i,h),(i,H)) : i \in V \setminus \{0\},\ 1 \leq h \leq H-1\}\}$

A node (i,h) is associated to node i being in layer h in the original graph (i.e., the path from node 0 to node i contains h arcs). Note that G_E is built by levels with the nodes of the original graph replicated H times. Consider the minimum directed Steiner tree problem in G_E with root node 0 and required node set $R = \{(i,H) : i \in V \setminus \{0\}\}$. Set the cost of arcs in A_0 and A_1 as equal to the cost of the corresponding edges in E, set the cost of arcs in A_2 as zero. It can be seen that a spanning tree with depth less or equal than H and cost C in the original graph corresponds to a Steiner tree in G_E, rooted at node 0 and containing all the required nodes, with the same cost. For example, if G is the graph depicted in the left of Figure 1, the corresponding layered graph G_E for $H = 3$ is shown in the left of Figure 2. The optimal HMST of G for $H = 3$ shown in the right of Figure 1 corresponds to the Steiner solution over G_E shown in the right of Figure 2.

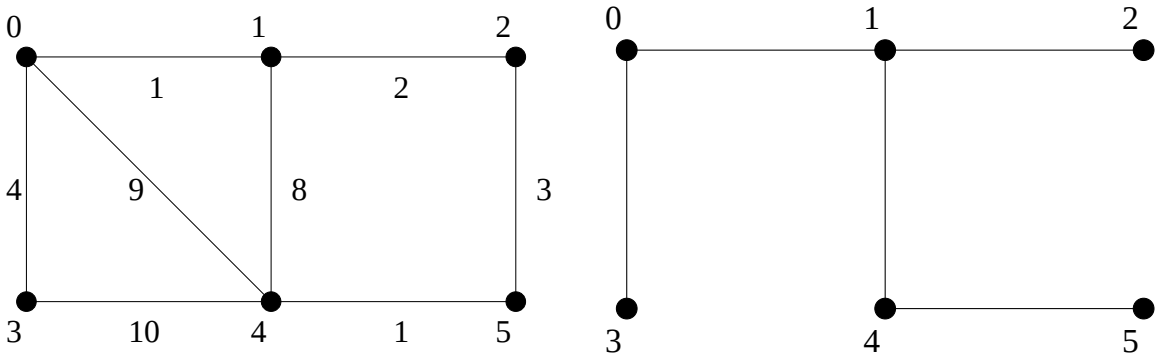

Figure 1: Example of a graph and its HMST for $H = 3$.

The interest of this construction is that one can use any model for the Steiner tree problem in the layered graph to provide a valid model for the HMST. Associate a binary variable X^h_{ij} to each arc $((i,h-1),(j,h))$ in A_E. Let $[V_E \setminus S, S]$ denote the set of arcs from $V_E \setminus S$ to S and $X^h[V_E \setminus S, S]$ denote the sum of the X^h_{ij} variables associated to the arcs on this cut. The following model, referred as the *hop-cut model*, is a particular case of the dicut Steiner model [10]:

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in E} c_{ij} \sum_{h=1}^{H} X^h_{ij} & & \\
\text{s.t.} \quad & \sum_{i \in V\setminus\{0,j\}} X^H_{ij} + \sum_{h=2}^{H} X^h_{jj} = 1 & & \forall j \in V \setminus \{0\} \\
& X^h[V_E \setminus S, S] \geq 1 & & \forall S;\ 0 \notin S \text{ and } S \cap R \neq \{\emptyset\} \\
& X^1_{0j} \in \{0,1\} & & \forall j \in V \setminus \{0\} \\
& X^h_{ij} \in \{0,1\} & & \forall (i,j) \in E;\ i \neq 0;\ h = 2,\ldots,H \\
& X^h_{jj} \in \{0,1\} & & \forall j \in V \setminus \{0\};\ h = 2,\ldots,H
\end{aligned}
$$

The hop-cut model has an exponential number of constraints and requires a cutting plane algorithm to be solved.

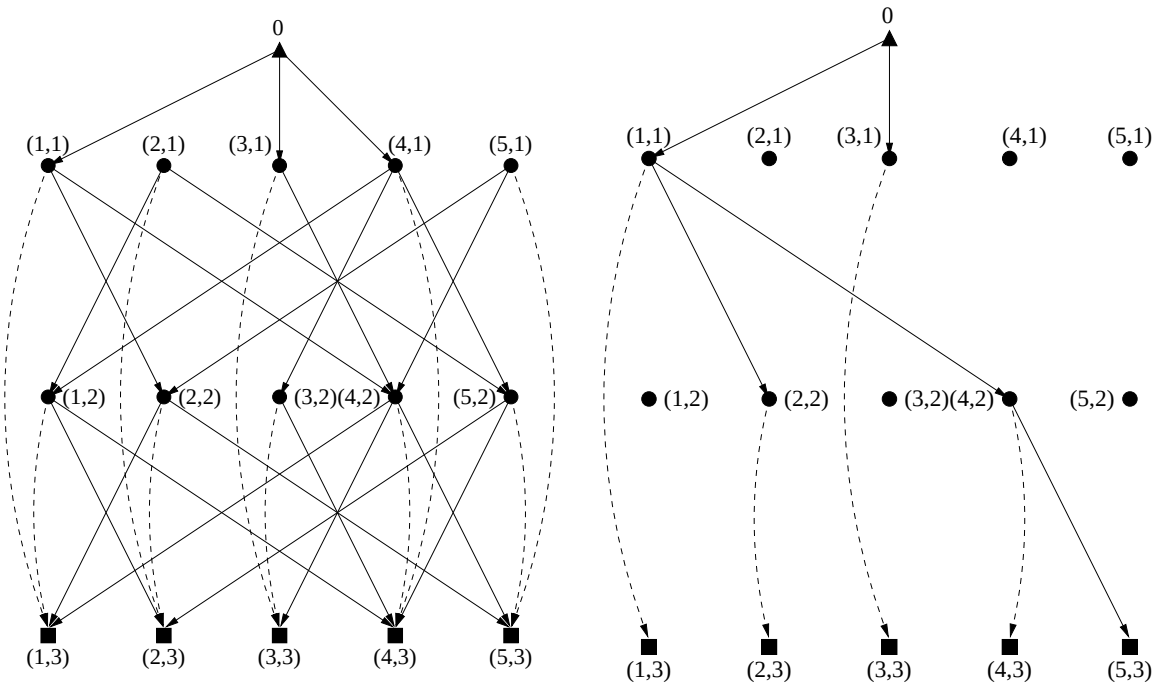

Figure 2: Layered graph for $H = 3$ and its optimal Steiner solution.

3. Cutting plane algorithm

The algorithm used to solve the hop-cut model follows closely the one given in Poggi de Aragão, Uchoa and Werneck [14] and described with more detail in [13]. It starts by applying a Dual Ascent heuristic [15]. This quickly provides a lower bound on the value of an optimal solution and also a good set of cut constraints to hot-start the cutting plane algorithm. The primal heuristic SPH-Prim [12], implemented as described in [4] is also run. This pair of lower and upper bounds may allow arc variables to be fixed by reduced costs. The cutting plane algorithm proceeds by separating rounds of violated cuts, using the algorithm of Hao and Orlin [9].

4. Computational Experiments

The experiments were run over in a machine with an Athlon XP 2.6 GHz processor and 1GB of RAM. The separation routine described in the previous section was embedded as a callback routine in the MIP solver provided by XPRESS 2005B package. The overall algorithm was tested on a set of 19 instances, the same ones as those used in the recent literature. Two families of instances come from a random placement of non-root vertices in a 100×100 grid. The root has a central position, coordinates $(50, 50)$, in the TC instances. The TE instances have the root in an eccentric position, coordinates $(0, 0)$. The instance costs correspond to truncated Euclidean distances. The third family of TR instances also correspond to complete graphs, but with edge costs randomly distributed. There are instances with 20, 30, 40, 50, 60, 80, 100, 120, and 160 non-root nodes. For each instance, we solved the HMST problem for $H = 3$, 4 and 5.

In order to reduce the size of each instance, we have used the following simple arc elimination test (see Gouveia [6]). 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 the transformation. Table 1 shows, for each instance, the percentage of number of arcs still remaining in each instance, after the elimination test was

performed. Note that the test is much more effective when applied to instances TC rather than to instances TE. This means that the reduced instances TE are larger than the reduced instances TC suggesting that the TE instances will be much more difficult to solve than the remaining instances.

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

Table 1: Remaining size of reduced HMST instances.

Tables 2 to 4 show the results obtained on TC, TR and TE instances. The first columns are the values of $|V|$ and H, next are the values of an optimal solution value and the values obtained by solving the linear relaxation of the HOP-CUT model. The solution of that linear relaxation was already integral on all instances tested (except on instance TE160 with $H = 5$, where the cutting plane algorithm failed to converge in reasonable time). Column HEU_DUAL gives the fast lower bound obtained by Dual Ascent, where HEU_PRIMAL is the fast upper bound got by the Prim-SPH. Column NCUTS is the number of cuts separated in the cutting plane algorithm. The last column is the total time to solve the instance.

$\|V\|$	H	OPT	LP	HEU_DUAL	HEU_PRIMAL	NCUTS	T(s)
	3	340	340	340	340	-	0
20	4	318	318	318	318	-	0.04
	5	312	312	312	312	-	0.03
	3	609	609	601	665	287	0.19
40	4	548	548	542	548	181	0.23
	5	522	522	522	524	294	0.4
	3	866	866	859	901	542	1.1
60	4	781	781	762	908	966	7.93
	5	734	734	726	876	903	6.13
	3	1072	1072	1064	1158	930	5.44
80	4	981	981	971	1028	2431	183
	5	922	922	918	954	2270	295
	3	1259	1259	1237	1361	2021	69.9
100	4	1166	1166	1151	1524	4327	1203
	5	1104	1104	1098	1366	5072	4052

Table 2: HMST results over TC instances.

$\|V\|$	H	OPT	LP	HEU_DUAL	HEU_PRIMAL	NCUTS	T(s)
	3	168	168	163	169	36	0.02
20	4	146	146	144	150	56	0.03
	5	137	137	137	137	-	0
	3	176	176	172	176	111	0.09
40	4	149	149	146	149	227	0.53
	5	139	139	137	140	278	0.54
	3	213	213	205	220	289	0.55
60	4	152	152	151	153	209	0.95
	5	124	124	124	124	-	0.2
	3	208	208	208	208	-	0.19
80	4	180	180	179	181	268	2.58
	5	164	164	164	167	769	13.1

Table 3: HMST results over TR instances.

A comparison with the results obtained by Dahl, Gouveia and Requejo [3] is shown in the last table. The first column identifies the type of instance (TC, TE or TR) and gives the number of nodes of the corresponding graph. The second column gives the value of H. In the third column, denoted by OPT we give the value of the corresponding optimal solution. The next columns compare LP bounds and times. When the time is given by a single value, this is the time to solve the linear relaxation, which is already integral. When there are two values following the format $a + b$, value a indicates the time needed to solve the linear programming relaxation while b indicates the additional time needed to obtain the integer optimum in a branching phase. The letter "m" indicates that the algorithm run out of memory.

$\|V\|$	H	OPT	LP	HEU_DUAL	HEU_PRIMAL	NCUTS	T(s)
	3	449	449	449	537	128	0.07
20	4	385	385	385	385	-	0.05
	5	366	366	359	390	335	0.65
	3	708	708	707	790	345	0.56
40	4	627	627	624	629	227	0.45
	5	590	590	589	638	940	22.25
	3	1525	1525	1519	1767	785	4.5
60	4	1336	1336	1328	1418	1475	65.6
	5	1225	1225	1219	1288	1482	137
	3	1806	1806	1802	1929	866	7.88
80	4	1558	1558	1549	1601	1409	50.1
	5	1442	1442	1434	1554	3852	2299
	3	2092	2092	2082	2329	1739	83
100	4	1788	1788	1771	2059	3256	1997
	5	1625	1625	1625	1751	1435	245
	3	1267	1267	1258	1734	1892	119
120	4	1074	1074	1071	1240	3822	4465
	5	969	969	962	1079	6455	27107
	3	1496	1496	1485	1822	2733	631
160	4	1229	1229	1219	1594	7632	47782
	5			1098	1262		

Table 4: HMST results over TE instances.

PROB, $\|V\|$	H	OPT	LP_HOP-CUT	T(s)	LP_DGR06	T(s)
	3	609	609	0.19	604.5	2+48
TC, 40	4	548	548	0.23	547	8+3
	5	522	522	0.4	522	13
	3	176	176	0.09	176	2
TR, 40	4	149	149	0.53	148.3	9+3
	5	139	139	0.54	139	26+1
	3	708	708	0.56	701.7	175+1984
TE, 40	4	627	627	0.45	625.3	969+9797
	5	590	590	22.25	588.1	2794+23052
	3	1072	1072	5.44	1069	164+2760
TC, 80	4	981	981	183	976.5	1824+25912
	5	922	922	295	920.6	4268+7729
	3	208	208	0.19	208	64+3
TR, 80	4	180	180	2.58	180	676+4
	5	164	164	13.1	164	1271+6
	3	1806	1806	7.88	1792.5	16519+146844
TE, 80	4	1558	1558	50.1	1544.5	154391+m
	5	1442	1442	2299		

Table 5: Comparing hop-cut and Dahl, Gouveia and Requejo [3] models.

References

[1] G. Dahl. The 2-hop spanning tree problem. *Operations Research Letters*, 23:21–26, 1998.

[2] G. Dahl, T. Flatbert, N. Foldnes, and L. Gouveia. The jump formulation for the hop-constrained minimum spanning tree problem. Technical report, Centro de Investigação Operacional, Faculdade de Ciências da Universidade de Lisboa, 2004.

[3] G. Dahl, L. Gouveia, and C. Requeijo. *On Formulations and Methods for the Hop-Constrained Minimum Spanning Tree Problem*, pages 493–515. Handbooks of Telecommunications. Springer, 2006.

[4] M. Poggi de Aragão and R. F. Werneck. On the implementation of MST-based heuristics for the Steiner problem in graphs. *Lecture Notes in Computer Science*, 2409:1–15, 2002.

[5] L. Gouveia. Using the Miller-Tucker-Zemlin constraints to formulate a minimal spanning tree problem with hop constraints. *Computers & Operations Research*, 22:959–970, 1995.

[6] L. Gouveia. Multicommodity flow models for spanning trees with hop constraints. *European Journal of Operational Research*, 95:178–190, 1996.

[7] L. Gouveia. Using variable redefinition for computing lower bounds for minimum spanning and Steiner trees with hop constraints. *INFORMS Journal on Computing*, 10:180–188, 1998.

[8] L. Gouveia and C. Requeijo. A new lagrangian relaxation approach for the hop-constrained minimum spanning tree problem. *European Journal of Operational Research*, 132:539–552, 2001.

[9] J. Hao and J. B. Orlin. A faster algorithm for finding the minimum cut of a graph. In *Proc. 3rd ACM-SIAM Symposium on Discrete Algorithms*, pages 165–174, 2002.

[10] N. Maculan. The Steiner problem in graphs. *Annals of Discrete Mathematics*, 31:185–212, 1987.

[11] P. Manyem and M. F. M. Stallmann. Some approximation results in multicasting. Technical Report TR-96-03, North Carolina State University, 21, 1996.

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

[13] E. Uchoa. *Algoritmos para Problemas de Steiner com Aplicações em Projeto de Circuitos VLSI*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro, 2001.

[14] E. Uchoa, M. Poggi de Aragão, and C. C. Ribeiro. Dual heuristics on the exact solution of large Steiner problems. *Electronic Notes in Discrete Mathematics*, 7:150–153, 2001.

[15] R. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical Programming*, 28:271–287, 1984.

[16] K. Woolston and S. Albin. The design of centralized networks with reliability and availability constraints. *Computers & Operations Research*, 15:207–217, 1988.