Solving Diameter Constrained Minimum Spanning Tree Problems in Dense Graphs

Andréa Cynthia dos Santos[1], Abilio Lucena[2] e Celso Carneiro Ribeiro[1]
[1] Departamento de Informática – PUC-Rio
Rua Marquês de São Vicente, 225, Gávea – 22453-900, Rio de Janeiro, RJ
[2] Departamento de Administração, UFRJ
Av. Pasteur 250 – 22290-240, Rio de Janeiro, RJ
{*cynthia, celso*}*@inf.puc-rio.br*
lucena@facc.ufrj.br

PUC-RioInf.MCC13/04 May, 2004

Abstract: In this study, a lifting procedure is applied to some existing formulations of the Diameter Constrained Minimum Spanning Tree Problem. This problem typically models network design applications where all vertices must communicate with each other at minimum cost, while meeting or surpassing a given quality requirement. An alternative formulation is also proposed for instances of the problem where the diameter of feasible spanning trees can not exceed given odd numbers. This formulation dominated their counterparts in this study, in terms of the computation time required to obtain proven optimal solutions. First ever computational results are presented here for complete graph instances of the problem. Sparse graph instances as large as those found in the literature were solved to proven optimality for the case where diameters can not exceed given odd numbers. For these applications, the corresponding computation times are competitive with those found in the literature.

Keywords: Diameter Constrained Minimum Spanning Tree Problem, lifting, network design applications, integer programming.

Resumo: Neste trabalho, um procedimento de lifting é aplicado em algumas formulações para o Problema da Árvore Geradora Mínima com Restrição de Diâmetro. Esse problema tipicamente modela aplicações em projetos de redes onde todos os nós devem comunicar-se a custo mínimo, mas garantindo um certo nível de serviço. Uma formulação alternativa é também proposta para instâncias do problema onde o diâmetro requerido da árvore geradora é ímpar. Esta formulação domina outras da literatura, em termos dos tempos de computação necessários para obter soluções ótimas. Os primeiros resultados computacionais para instâncias de grafos completos na literatura são apresentados neste trabalho. Instâncias de grafos esparsos maiores que as encontradas na literatura foram resolvidas para o caso onde o diâmetro requerido é ímpar. Neste caso, os tempos computacionais são competitivos com aqueles encontrados na literatura.

Palavras-chave: Problema da Árvore Geradora Mínima com Restrição de Diâmetro, lifting, projeto de redes de computadores, programação inteira.

1 Introduction

Let $G = (V, E)$ be a finite undirected connected graph with a set V of vertices and a set E of edges. Assume that a cost c_{ij} is associated with every edge $[i, j] \in E$, with $i < j$. Denote by $T = (V, E')$ a spanning tree of G, with $E' \subseteq E$. For every pair of distinct vertices $i, j \in V$, there exists a unique path $\mathcal{P}_{ij}$ in T linking i and j. Denote by d_{ij} the number of edges in $\mathcal{P}_{ij}$ and by $d = \max\{d_{ij} : i, j \in V\}$ the *diameter* of T. Given a positive integer $2 \leq D \leq |V| - 1$, the Diameter Constrained Minimum Spanning Tree Problem (DCMST) is to find a minimum cost spanning tree T with $d \leq D$.

DCMST has been shown to be NP-hard when $D \geq 4$ [6]. The problem typically models network design applications where all vertices must communicate with each other at minimum cost, while meeting or surpassing a given quality requirement [7]. Additional applications are found in data compression [3] and distributed mutual exclusion in parallel computing [4, 11].

DCMST formulations in the literature implicitly use a property of feasible diameter constrained spanning trees, pointed out by Handler [8]. Consider first the case where D is even. Handler noted that a *central vertex* $i \in V$ must exist in a feasible tree T, such that no other vertex of T is more than $D/2$ edges away from i. Conversely, if D is odd, a *central edge* $e = [i, j] \in E$ must exist in T, such that no vertex of T is more than $(D-1)/2$ edges away from the closest extremity of (i, j). Another feature shared by these formulations is that, in addition to the use of natural space variables (i.e. variables associated with the edges of G, for this application), the central vertex (resp. edge) property of T is enforced through the use of an auxiliary network flow structure. In doing so, connectivity of T is naturally enforced by these structures.

The formulation proposed in [1, 2] for even D relies on an artificial vertex to model central spanning tree vertices. For odd D, however, the corresponding formulation in [1, 2] do not use either artificial vertices or edges. Similarly, formulations in [7], irrespective of D being odd or even, do not rely on artificial vertices or edges. Another distinction between formulations in [1, 2] and those in [7] is that the former contains multicommodity network flow structures, while the latter contains single commodity network flow ones. As a result, tighter linear programming relaxations are obtained in [7], albeit at a much larger computer memory requirement.

Achuthan et al. [1, 2] do not present computational results for their DCMST formulation. Gouvea and Magnanti [7] used the Mixed Integer Programming (MIP) solver CPLEX 5.0 to test their formulation uniquely on fairly sparse graphs.

In this paper, we introduce an alternative form of enforcing the central edge property for the odd D case of DCMST. The proposed model is based on the use of an artificial vertex. We also apply a lifting procedure to strengthen the formulations in [1, 2]. Original formulations and lifted versions of them were tested, under the MIP solver CPLEX 9.0, on complete graph instances as well as on sparse graph ones. For the computational results obtained, lifted versions of the formulations invariably required significantly less computation time to

prove optimality than their unlifted counterparts. That feature was further enhanced for the odd D case, with the use of the artificial vertex model.

In Section 2, a summary of the main results for formulations in [1, 2] is presented. In Section 3, the artificial vertex DCMST formulation for odd D is described. Strengthened (i.e. lifted) versions of the formulations in [1, 2] are presented in Section 4. Computational experiments for dense and sparse graph instances are reported in Section 5. In these experiments, denser instances than previously attempted in the literature were solved to proven optimality. Concluding remarks are made in Section 6.

2 Formulations

Formulations in this study make use of a directed graph $G' = (V, A)$. Graph G' is obtained from the original undirected graph $G = (V, E)$, as follows. For every edge $e = [i, j] \in E$, with $i < j$, there exist two arcs (i, j) and $(j, i) \in A$, with costs $c_{ij} = c_{ji}$. Let $L = D/2$ if D is even and $L = (D - 1)/2$, otherwise.

The very first formulations for DCMST were proposed by Achuttan et al. [1, 2]. Distinct formulations are presented by the authors for even D and odd D cases of the problem. Consider first the case where D is even and introduce an artificial vertex, denoted by r, into G'. Let $G'' = (V', A')$ be the resulting graph with $V' = V \cup \{r\}$ and $A' = A \cup \{(r, 1), \ldots, (r, |V|)\}$. Associate a binary variable x_{ij} with every arc $(i, j) \in A'$ and a non-negative variable u_i with every vertex $i \in V'$. Binary variables x_{ij} are used to identify a spanning tree, while variable u_i denotes the number of arcs in a path from r to $i \in V$. For even D, DCMST is formulated as follows:

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

$$\sum_{j \in V} x_{rj} = 1 \tag{2}$$

$$\sum_{(i,j)\in A'} x_{ij} = 1 \quad \forall j \in V \tag{3}$$

$$u_i - u_j + (L+1)x_{ij} \leq L \quad \forall (i,j) \in A' \tag{4}$$

$$x_{ij} \in \{0, 1\} \quad \forall (i,j) \in A' \tag{5}$$

$$0 \leq u_i \leq L + 1 \quad \forall i \in V'. \tag{6}$$

Equation (2) ensures that the artificial vertex r is connected to exactly one vertex in V, i.e. the central spanning tree vertex. Constraints (3) establish that exactly one arc must be incident to each vertex of V. Constraints (4) and (6) ensure that paths from the artificial vertex r to each vertex $i \in V$ have at most $L + 1$ arcs. Constraints (5) are the integrality

requirements. Edges $[i,j] \in E$ such that $x_{ij} = 1$ or $x_{ji} = 1$ in a feasible solution to (2)-(6) define a spanning tree T of G with diameter less than or equal to D.

We now consider the odd D case of DCMST. Let z_{ij} be a binary variable associated with each edge $[i,j] \in E$, with $i < j$. Whenever $z_{ij} = 1$, edge $[i,j]$ is selected as the central spanning tree edge. Otherwise, $z_{ij} = 0$. For D odd, DCMST is formulated as follows:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} + \sum_{[i,j]\in E} c_{ij}z_{ij} \tag{7}$$

$$\sum_{[i,j]\in E} z_{ij} = 1 \tag{8}$$

$$\sum_{(i,j)\in A} x_{ij} + \sum_{[i,j]\in E} z_{ij} + \sum_{[j,i]\in E} z_{ji} = 1 \quad \forall j \in V \tag{9}$$

$$u_i - u_j + (L+1)x_{ij} \leq L \quad \forall (i,j) \in A \tag{10}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{11}$$

$$z_{ij} \in \{0,1\} \quad \forall [i,j] \in E \tag{12}$$

$$0 \leq u_i \leq L \quad \forall i \in V. \tag{13}$$

Equation (8) ensures that there must be exactly one central edge. Constraints (9) establish that for any vertex $i \in V$ either there is an arc incident to it or else vertex i must be one of the extremities of the central spanning tree edge. Constraints (10) and (13) ensure that spanning tree paths from the closest extremity of the central edge to every other vertex $i \in V$ have at most L arcs. Constraints (11) and (12) are the integrality requirements. In a feasible solution to (8)–(13), the central edge together with those edges $[i,j] \in E$ such that $x_{ij} = 1$ or $x_{ji} = 1$ define a spanning tree T of G with diameter less than or equal to D.

3 An alternative formulation for the odd D case

The formulation in [1, 2] for D odd selects one edge in E to be central and simultaneously builds an auxiliary network flow problem around that edge. Flow emanating from the central edge is then controlled to enforce the diameter constraint. Figure 1 (a) illustrates a solution obtained for $D = 3$. Notice that edge $[p,q]$ plays the central edge role and that any spanning tree leaf is no more than $L = (D-1)/2 = 1$ edges away from edge $[p,q]$.

An alternative formulation which uses an artificial vertex r, as for the even D case, is also possible here. Recall that, for D even, the artificial vertex r is connected to exactly one vertex of V, i.e. the central spanning tree vertex. Now, the artificial vertex r will be connected to exactly two vertices. Namely those two vertices incident on the central edge. This situation is modeled by implicitly enforcing selection of a central edge $[p,q] \in E$ by explicitly forcing *artificial* edges $[p,r]$ and $[q,r]$ to appear in the solution. An illustration

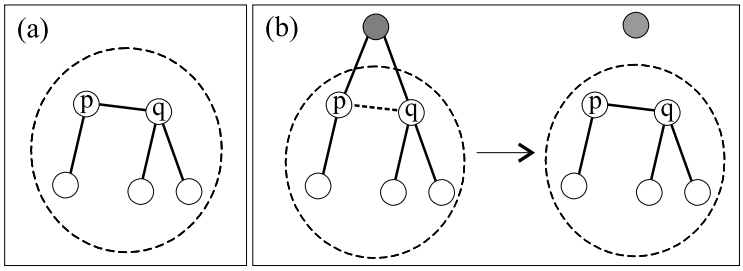

Fig. 1. Solutions to DCMST in the odd case with $D = 3$.

of this scheme appears in Figure 1 (b). A feasible spanning tree T of G is obtained by eliminating the two edges incident on r and connecting their extremities through the central edge $[p, q]$.

The motivation behind our formulation for D odd is to highlight a structure that has already been well studied from a polyhedral viewpoint. In doing so, we expect to strengthen the overall DCMST formulation through the use of facet defining inequalities for that structure.

Consider the same notation and variables introduced in Section 2 for D odd. Additional variables x_{ri}, for every $i \in V$, are introduced to represent the edges associated with artificial vertex r. An edge $[i, j] \in E$ is selected as the central spanning tree edge if and only if edges $[r, i]$ and $[r, j]$ are also selected. This condition is enforced through the nonlinear equation $z_{ij} = x_{ri} \cdot x_{rj}$ or convenient linearizations of it. A valid formulation for DCMST when D is odd is given by:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} + \sum_{[i,j]\in E} c_{ij}z_{ij} \tag{14}$$

$$\sum_{j\in V} x_{rj} = 2 \tag{15}$$

$$\sum_{(i,j)\in A'} x_{ij} = 1 \quad \forall j \in V \tag{16}$$

$$\sum_{[i,j]\in E} z_{ij} = 1 \tag{17}$$

$$z_{ij} \geq x_{ri} + x_{rj} - 1 \quad \forall [i,j] \in E \tag{18}$$

$$z_{ij} \leq x_{ri} \quad \forall [i,j] \in E \tag{19}$$

$$z_{ij} \leq x_{rj} \quad \forall [i,j] \in E \tag{20}$$

$$u_i - u_j + (L+1)x_{ij} \leq L \quad \forall (i,j) \in A' \tag{21}$$

$$0 \leq u_i \leq L+1 \quad \forall i \in V \tag{22}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A' \tag{23}$$

$$z_{ij} \in \{0,1\} \quad \forall [i,j] \in E. \tag{24}$$

Equation (15) establishes that the artificial central vertex r is connected to exactly two vertices of V. Constraints (16) establish that there is exactly one arc incident to each vertex of V. Constraints (17) to (20) give a linearization of $z_{ij} = x_{ri} \cdot x_{rj}$ for every edge $[i,j] \in E$. Finally, constraints (21) and (22) ensure that the paths from the artificial vertex r to each vertex $i \in V$ have at most $L+1$ arcs. Constraints (23) and (24) are the integrality requirements.

We now derive valid inequalities for the formulation (14)-(24). If constraint (15) is multiplied by variable x_{ri}, for $i \in V$,

$$\sum_{j \in V, j \neq i} x_{ri} \cdot x_{rj} + x_{ri} \cdot x_{ri} = 2 \cdot x_{ri} \tag{25}$$

results. Bearing in mind that all variables in (25) are binary 0-1 and consequently $x_{ri} = x_{ri} \cdot x_{ri}$ holds, it is valid to write

$$\sum_{j \in V, j \neq i} x_{ri} \cdot x_{rj} = x_{ri}. \tag{26}$$

However, since $z_{ij} = x_{ri} \cdot x_{rj}$ for every edge $[i,j] \in E$ and x_{ri} and x_{rj} cannot simultaneously be equal to 1 if an edge $[i,j]$ does not exist in E, valid constraints for (14)–(24) are

$$\sum_{[i,j] \in E} z_{ij} + \sum_{[j,i] \in E} z_{ji} = x_{ri} \quad \forall i \in V. \tag{27}$$

Constraints (27) are redundant for formulation (14)–(24) but are not necessarily so for its linear programming relaxation.

Additional valid inequalities for (14)–(24) can be found if one concentrates on inequalities (18)–(20) and the underlying Boolean quadric polytope [10].

4 Lifting

In this section, following the work of Desrochers and Laporte [5], we lift the Miller-Tucker-Zemlin [9] inequalities $u_i - u_j + (L+1)x_{ij} \leq L, \forall (i,j) \in A'$. In doing so, strengthened versions of DCMST formulations presented in previous sections are obtained. The idea of lifting consists in adding a valid nonnegative term $\alpha_{ji}x_{ji}$ to the above inequalities, transforming them into

$$u_i - u_j + (L+1)x_{ij} + \alpha_{ji}x_{ji} \leq L. \tag{28}$$

The larger is the value of α_{ji}, the larger will be the reduction in the original solution space. If $x_{ji} = 0$, then α_{ji} may take any value. Suppose now $x_{ji} = 1$. Then, $u_i = u_j + 1$

since the path from the central vertex in the even case (resp. from the closest extremity of the central edge in the odd case) to vertex $i \in V$ visits j before visiting i. Moreover, $x_{ji} = 1$ implies $x_{ij} = 0$ due to constraints (3) or (9). By substitution in (28), we obtain $u_i - u_j + (L+1)x_{ij} + \alpha_{ji}x_{ji} \leq L \Rightarrow 1 + u_j - u_j + \alpha_{ji} \leq L \Rightarrow \alpha_{ji} \leq L - 1$. To maximize the value of α_{ji}, we take $\alpha_{ji} = L - 1$. Then,

$$u_i - u_j + (L+1)x_{ij} + (L-1)x_{ji} \leq L \tag{29}$$

is a valid inequality for all $(i,j) \in A'$ (resp. for all $(i,j) \in A$) for $D > 2$ in the even case (resp. for $D > 3$ in the odd case).

We now derive improved generalized upper bounds for the variables u_i, for $i \in V$. In the even case, there is an artificial vertex r such that $u_r = 0$. The central vertex connected to r will necessarily be the first vertex to be visited in any path emanating from r. Then,

$$u_i \leq (L+1) - Lx_{ri} \quad \forall i \in V. \tag{30}$$

Moreover, $u_i \leq L$ for any vertex $i \in V$ which is not a leaf of the spanning tree. Then,

$$u_i \leq (L+1) - x_{ij} \quad \forall (i,j) \in A \tag{31}$$

holds.

We now consider the odd case. If an edge $[i,j] \in E$ is the central one, then $z_{ij} = 1$, $u_i = 0$, and $u_j = 0$. In consequence,

$$u_i \leq L - Lz_{ij} \quad \forall [i,j] \in E. \tag{32}$$

Analogously to the odd case, $u_i < L$ for any vertex $i \in V$ which is not a leaf of the spanning tree. Then,

$$u_i \leq L - x_{ij} \quad \forall (i,j) \in A. \tag{33}$$

Inequalities (30) and (31) define improved generalized upper bounds for the even D case, while inequalities (32) and (33) correspond to new generalized upper bounds for the odd S case.

We now derive improved generalized lower bounds for the variables u_i, for $i \in V$. In the even case, $u_i \geq 1 \geq x_{ri}$ for any vertex $i \in V$. If i is not directly connected to the central vertex, then $x_{ri} = 0$ and $u_i \geq 2$. If these two conditions are taken into account simultaneously, we have the first improvement in the lower bounds:

$$u_i \geq x_{ri} + 2 \sum_{j \in V : j \neq r} x_{ji} \quad \forall i \in V. \tag{34}$$

The above condition is simpler in the odd case, where no central vertex exists:

$$u_i \geq \sum_{j \in V, j \neq i} x_{ji} \quad \forall i \in V. \tag{35}$$

5 Computational results

Computational experiments were performed on a Pentium IV machine with a 2.0 GHz clock and 512 MB of RAM memory, using MIP solver CPLEX 9.0 under default parameters. In these experiments, the alternative formulation proposed for the odd D case of DCMST was reinforced with valid inequalities (27).

Test instances were generated as follows. For a graph with a number $n = |V|$ of vertices, the uniform distribution was used to draw n points with integer coordinates in a square of sides 100 on the Euclidean plane. Vertices were associated to points and edge costs were taken as the truncated Euclidean distance between corresponding pairs of points. Sparse graph instances with $m = |E|$ edges were generated as in [7]. The minimum cost spanning star is computed first and all of its $n-1$ edges are selected. The remaining $m-n-1$ edges are taken as the least cost edges not already contained in the minimum cost star. In all 19 odd D instances and 18 even D instances were generated. For each of the two cases, 12 complete graph instances (with up to 25 vertices) were generated. Test instance details are summarized in Tables 1 and 2.

Table 1 gives numerical results for odd D instances. For each instance, the number of vertices, the number of edges, and the value of the diameter D are given. These entries are followed by the results obtained with the original formulation in [1, 2] (A), the original formulation with lifting (B), and the new artificial central vertex formulation with lifting (C). For each formulation, the CPU time required to prove optimality is given in seconds together with the number of nodes visited in the branch-and-bound tree. Table 2 gives the same results for the even D case, except for the central vertex formulation which does not apply in this case.

In spite of the considerable duality gaps associated with the formulations tested here, the lifted formulations we suggest are capable of solving, to proven optimality, sparse instances as large as those found in the literature in competitive CPU times.

No results appear in the literature for complete graph DCMST instances. A possible explanation for that is the large computer memory demands required by the other existing DCMST formulations [1, 2]. The very first computational results for complete graph DCMST instances are thus introduced in this study.

¿From the computational results presented, it should also be noticed that our alternative odd D case formulation dominates their counterparts in this study in terms of the CPU time required to prove optimality.

6 Conclusions

In this study, DCMST formulations proposed in [1, 2] were strengthened through the use of a lifting procedure. In doing so, substantial duality gap reductions were attained for the computational experiments carried out. Additionally, we also propose an artificial central

Table 1. Numerical results for the odd D case.

			(A)		(B)		(C)	
$\|V\|$	$\|E\|$	D	time (s)	nodes	time (s)	nodes	time (s)	nodes
10	45	5	0.14	85	0.16	62	0.13	28
10	45	7	0.14	190	0.17	88	0.17	49
10	45	9	0.05	40	0.10	73	0.08	10
15	105	5	40.95	73331	22.27	18640	31.22	21814
15	105	7	85.13	163355	25.02	23226	19.23	16118
15	105	9	76.14	146948	10.56	9597	7.80	6174
20	190	5	1686.17	1753713	272.00	132678	216.36	95813
20	190	7	31.16	36168	8.39	3532	5.05	1519
20	190	9	884.36	982551	151.5	85204	91.33	46559
25	300	5	–	–	73857.81	24309253	51551.80	17235497
25	300	7	24513.01	15083987	20160.13	6225967	16617.61	5272473
25	300	9	177024.22	66213391	9172.04	2438057	40183.44	12040315
20	50	5	39.78	93283	7.74	9386	5.58	3707
20	50	7	39.06	68843	3.16	4908	2.23	1915
20	50	9	70.12	128376	26.16	45561	40.28	57603
40	100	5	3596.34	3506756	780.74	688912	20.67	13993
40	100	7	12581.45	12272065	976.98	849861	207.64	171846
40	100	9	27735.56	23763469	4584.19	3530233	23359.05	15583400
60	150	5	–	–	215161.75	116775357	11644.75	5003876

Table 2. Numerical results for the even D case.

			(A)		(B)	
$\|V\|$	$\|E\|$	D	time (s)	nodes	time (s)	nodes
10	45	4	0.95	1849	0.77	1300
10	45	6	0.13	55	0.08	7
10	45	10	0.06	29	0.08	17
15	105	4	65.8	73024	24.17	26711
15	105	6	53.19	66834	32.53	41882
15	105	10	38.41	61822	8.95	11790
20	190	4	7462.1	4877014	1888.02	1091803
20	190	6	1630.58	1210813	593.91	412770
20	190	10	2729.48	2285819	172.81	144382
25	300	4	–	–	158836.45	64913343
25	300	6	43044.61	17498605	5194.90	2119161
25	300	10	1031.36	565737	459.88	205747
20	50	4	62.47	115144	0.64	615
20	50	6	221.31	446477	10.81	16396
20	50	10	619.52	1443014	74.15	173234
40	100	4	8957.38	8110166	54.14	51476
40	100	6	205940.95	167119305	909.95	1012212
40	100	10	–	–	146019.52	155646590

vertex strategy for modeling the odd D case of the problem. For the computational tests carried out, the new formulation dominated its odd D counterparts in in terms of total CPU time required to prove optimality. The same idea could also be extended to other existing formulations such as those presented in [7].

For sparse graphs instances, the strongest model proposed in this study was capable of solving, to proven optimality, instances as large as those previously solved in the literature [7]. It is worth mentioning here that the models suggested by Gouvea and Magnanti typically produce very small duality gaps. However, they are quite demanding in terms of computer memory requirements (particularly the model involving variables with four indices). In consequence, they do not appear adequate to directly tackling dense graph instances of the problem.

In spite of the considerable duality gaps observed in our computational experiments, our approach was capable of solving, to proven optimality, complete graph instances with up to 25 vertices. These are the first results ever presented for dense graph DCMST instances.

We conclude by pointing out that the alternative odd D formulation introduced here can be further strengthened with valid inequalities associated with the Boolean quadric polytope.

References

1. N.R. Achuthan, L. Caccetta, P.A. Caccetta, and J.F. Geelen. Algorithms for the minimum weight spanning tree with bounded diameter problem. In K.H. Phua, C.M. Wand, W.Y. Yeong, T.Y. Leong, H.T. Loh, K.C. Tan, and F.S. Chou, editors, *Optimisation Techniques and Applications*, volume 1, pages 297–304. World Scientific, 1992.
2. N.R. Achuthan, L.Caccetta, P.A. Caccetta, and J.F. Geelen. Computational methods for the diameter restricted minimum weight spanning tree problem. *Australasian Journal of Combinatorics*, 10:51–71, 1994.
3. A. Bookstein and S.T. Klein. Compression of correlated bitvectors. *Information Systems*, 16:110–118, 2001.
4. N. Deo and A. Abdalla. Computing a diameter-constrained minimum spanning tree in parallel. In G. Bongiovanni, G. Gambosi, and R. Petreschi, editors, *Algorithms and Complexity*, volume 1767, pages 17–31. 2000.
5. M. Desrochers and G. Laporte. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10:27–36, 1991.
6. M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-Completeness*. W.H. Freeman, New York, 1979.
7. L. Gouveia and T.L. Magnanti. Modelling and solving the diameter-constrained minimum spanning tree problem. Technical report, DEIO-CIO, Faculdade de Ciências, 2000.
8. G.Y. Handler. Minimax location of a facility in an undirected graph. *Transportation Science*, 7:287–293, 1978.
9. C.E. Miller, A.W. Tucker, and R.A. Zemlin. Integer programming formulations and traveling salesman problems. *Journal of the ACM*, 7:326 – 329, 1960.
10. M. Padberg. The boolean quadric polytope: Some characteristics and facets. *Mathematical Programming*, 45:139–172, 1988.
11. K. Raymond. A tree-based algorithm for distributed mutual exclusion. *ACM Transactions on Computers*, 7:61–77, 1989.