COMPUTATIONAL METHODS FOR THE DIAMETER RESTRICTED MINIMUM WEIGHT SPANNING TREE PROBLEM

N.R. Achuthan, L. Caccetta, P. Caccetta and J.F. Geelen
School of Mathematics and Statistics
Curtin University of Technology
GPO Box U1987
Perth, 6001
Western Australia.

ABSTRACT:

Let G be a simple undirected graph with non-negative edge weights. In this paper we consider the following combinatorial optimization problem : Find, in G, a minimum weight spanning tree having diameter at most D. This problem is trivial for $D \leq 3$ and NP-complete for $D \geq 4$. In this paper we develop and implement a number of Branch and Bound algorithms for this problem. Computational results, based on simulated problems, are discussed.

1. INTRODUCTION

Let G = (V,E) denote a finite undirected simple graph with vertex set V and edge set E. We assume that G is connected and every edge (x,y) has a non-negative weight w(x,y). Determining a **minimum weight spanning tree (MWST)** in G is a fundamental problem that arises in network design and as a subproblem in many combinatorial optimization problems such as vehicle routing. Very efficient procedures for solving the MWST problem exist [5]. In many applications, one is interested in determining a minimum weight spanning tree having certain prescribed properties. Except for trivial restrictions such

Australasian Journal of Combinatorics 10(1994), pp.51-71

problems are computationally difficult [8]. In this paper we consider the case when the spanning tree has a diameter restriction. The **distance** $d(x,y)$ between two vertices x and y is the number of edges in the shortest (x,y) - path in G (note that the shortest is in terms of the number of edges). The **diameter** $d(G)$ of G is defined as the maximum distance in G. The **minimum weight spanning tree with bounded diameter D (MWST-D) problem** is :

Find, in a given weighted graph G, a minimum weight spanning tree of diameter at most D.

Garey and Johnson [6] have shown that the MWST-D problem is NP-complete for any fixed $D \geq 4$; the problem is trivial for $D \leq 3$. As this problem arises in network applications [3] it is of interest to develop both exact and heuristic algorithms to solve it.

Achuthan and Caccetta [1] provided a mixed integer linear programming formulation (MILP) of the MWST-D problem. In [2] we gave another MILP formulation of this problem as well as a number of solution procedures based on Branch and Bound methods. A comparative analysis, based on simulated problems, of these methods indicated significant computational advantage could be achieved by exploiting the "level structure" of a diameter restricted tree. The objective of this paper is to generalize some of the ideas introduced in [2]. As we shall see in Section 4 this results in significantly improved algorithms.

In the next section we give an improved MILP formulation of the MWST-D problem. Section 3 summarizes the procedures described in [2] which are used as a basis of our work in Section 4.

2. MILP FORMULATION OF THE MWST-D PROBLEM

MILP formulations of the MWST-D problem have been proposed in [1,2]. These formulations extend the given weighted graph into a directed graph and make use of some of the ideas associated with the travelling salesman problem (TSP) and the vehicle routing problem (VRP). Consequently solution procedures for the TSP [7] and VRP can be utilized to solve the MWST-D problem. To date, our computational experimentation suggests that the application of standard MILP solution techniques yields efficient solutions only for relatively small problems. However, exploiting the structure of the MILP formulation could result in more efficient procedures. Our objective in this paper is to develop solution procedures based on the level structure of a diameter restricted tree. We present in this section a MILP formulation for the MWST-D problem which is a little simpler than the previous formulations given. We give separate formulations according to the parity of D.

Formulation D even :

For convenience we let $D = 2L$ and $V(G) = \{1,2,\ldots,n\}$. We extend the given graph $G = (V,E)$ to a directed graph $G' = (V',\mathcal{A}')$ by replacing each edge of G by a pair of oppositely directed arcs each having weight equal to that of the edge and adding a new vertex s (the source) and joining it to every vertex of G by an arc having zero weight. Writing w_{ij} for $w(i,j)$, our MILP formulation is :

$$\text{Minimize } f = \sum_{(i,j)\in\mathcal{A}'} w_{ij}\, x_{ij} \tag{2.1}$$

subject to

$$\sum_{j=1}^{n} x_{sj} = 1 \ , \tag{2.2}$$

$$\sum_{\substack{i \in V' \\ i \neq j}} x_{ij} = 1 \ , \qquad \text{for each } j \in V \tag{2.3}$$

$$y_i - y_j + (L + 1)x_{ij} \le L, \text{ for each } (i,j) \in \mathcal{A}' \tag{2.4}$$

$$x_{ij} = 0 \text{ or } 1 \ , \qquad \text{for each } (i,j) \in \mathcal{A}' \tag{2.5}$$

and

$$0 \le y_i \le L+1 \qquad \text{for each } i \in V' \tag{2.6}$$

Arguments similar to those used in [1] will establish that the above formulation does indeed solve the MWST-D problem for even D. In brief, one proceeds as follows.

Consider a solution $[x_{ij}, y_i]$ to the constraints (2.2) to (2.6). Restriction to the arcs of G′ with $x_{ij} = 1$ gives rise to a directed graph G* having the following properties. The vertex s has, by (2.2), outdegree 1 and every vertex of V′\{s} has, by (2.3), indegree 1. Condition (2.4) ensure that G* has no cycles. Thus G* consists of directed paths from source s. Further, (2.4) and (2.6) ensure that each such path has length at most L+1. Hence from G* - s we have a tree (ignoring directions) of diameter at most D. Conversely, given any spanning tree of G with diameter at most D we can easily construct (see [1]) a feasible solution (x_{ij}, y_i) to the MILP problem (2.1) to

(2.6). Hence, the MILP (2.1) to (2.6) solves the MWST-D problem.

Formulation D odd:

Let $D = 2L + 1$ and $V(G) = \{1,2,\dots,n\}$. We form the directed graph $G' = [V,\mathcal{A}]$ from $G = [V,E]$ by replacing each edge of G by a pair of oppositely directed arcs each having weight equal to that of the edge. Note that unlike the even D case we do not add any additional vertices. With each arc (i,j) of $\mathcal{A}$ we associate a 0-1 variable x_{ij} and with edge (ℓ,k) of E we associate a 0-1 variable $z_{\ell k}$. Our formulation is :

$$\text{Minimize} \quad f = \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \sum_{(i,j)\in E} c_{ij}z_{ij} \tag{2.7}$$

subject to

$$\sum_{(i,j)\in E} z_{ij} = 1 \tag{2.8}$$

$$\sum_{\substack{i\\(i,j)\in\mathcal{A}}} x_{ij} + \sum_{\substack{i\\(i,j)\in E}} z_{ij} = 1, \quad \text{for each } j \in V. \tag{2.9}$$

$$y_i - y_j + (L+1)x_{ij} \le L, \quad \text{for each } (i,j) \in \mathcal{A} \tag{2.10}$$

$$x_{ij} = 0 \text{ or } 1, \quad \text{for each } (i,j) \in \mathcal{A} \tag{2.11}$$

$$z_{ij} = 0 \text{ or } 1, \quad \text{for each } (i,j) \in E \tag{2.12}$$

and

$$0 \leq y_i \leq L, \qquad \text{for each } i \in V. \qquad (2.13)$$

An odd diameter tree T can be put in a layer structure with a root edge e = (u,v) such that all directed paths have u or v as their origin. The above formulation captures this structure; the z_{ij}'s are used to identify the root edge. The justification of the formulation (2.7) to (2.13) is similar to that of the even case.

3. KNOWN SOLUTION PROCEDURES

Motivation for the new procedures developed in the next section comes from the limitation of the Branch and Bound methods introduced in [2]. In this section we briefly review these methods; we assume basic familiarity with Branch and Bound methods [7]. In brief, the Branch and Bound method for an optimization problem involves the decomposition of the given problem into a number of smaller sized subproblems. The important components of this procedure are branching, bounding and searching strategies. The branching strategy dictates the manner in which a given problem is decomposed into subproblems. In these algorithms the subproblems are MWST-D problems in a subgraph of the given weighted graph G with the added requirement that some edges are included and some are excluded from the spanning tree. A solution to a relaxation of the subproblem provides a lower bound on the objective function value. Here the relaxed (the diameter restriction dropped) subproblems are MWST-problems with specified included and excluded edges.

Branching Rule 1 : If the solution to the relaxed subproblem is not diameter feasible, then the resulting tree contains a path of length D + 1 and a further D subproblems are generated by the inclusion and exclusion of the edges of this path.

For our second branching rule we make use of the following notation. Let P denote the subproblem and G_p the corresponding graph. Note that P is a MWST-D problem on G_p with edge restrictions.

Branching Rule 2: If the solution T_p to the relaxed subproblem is not diameter feasible, and there exists a diameter feasible tree T'_p in G_p not containing any of the edges excluded by the subproblem, then two further subproblems are generated by the inclusion and exclusion of an edge in T'_p which is not in T_p.

Tables 3.1 and 3.2 give computational results obtained by the implementation of the above branching rules on a SUN SPARC Workstation operating at 28.5 MIPS. For each case we tested 50 complete graphs with randomly generated edge-weights in the interval [1,1000]. We use the following heuristic to determine the initial upper bound. Further, our heuristic can be used with any known procedure for finding a diameter feasible spanning tree; we use the breadth first search method from each vertex.

Heuristic : Find a diameter feasible spanning tree T in the graph G, if one exists. For each edge e, chosen in order of decreasing weight, perform the following :

if G - {e} contains a diameter feasible tree T′ then redefine T := T′ and G := G - {e}.

Of a number of heuristics tested the above simple procedure performed competitively in terms of running time and the objective function value of the diameter feasible tree generated. We note that in our implementation of Branching Rule 2 we apply the Heuristic to $|V|$ diameter feasible trees generated by specifying the root vertex. This provides us with a good upper bound as well as an ordering (based on the weight of the tree) of the vertices for processing as roots. Tables 3.1 and 3.2 give the computational results for the two branching rules.

$\|V\|$	D	CPU TIME (SEC) Ave	Min	Max	No. of Subproblems Ave	Min	Max
10	8	0.0	0.0	0.0	2	0	27
	7	0.0	0.0	0.1	4	0	49
	6	0.0	0.0	0.1	11	0	165
	5	0.1	0.0	0.1	41	0	324
	4	0.1	0.0	0.4	201	0	1167
15	8	0.1	0.0	0.4	24	0	329
	7	0.3	0.1	0.9	52	0	478
	6	0.2	0.0	1.0	180	0	970
	5	2.7	0.2	24.2	1801	0	12586
	4	82.1	0.1	1440.8	30611	42	339073
20	8	0.4	0.1	3.0	121	0	1933
	7	2.4	0.5	10.9	436	0	3969
	6	7.5	0.1	92.9	3114	23	29846

Table 3.1 : Branching Rule 1 (Breadth first search)

$\|V\|$	D	CPU TIME Ave	(sec) Min	Max	No. of Ave	Subproblems Min	Max
10	8	0.0	0.0	0.0	7	0	158
	7	0.0	0.0	0.2	95	0	850
	6	0.0	0.0	0.1	33	0	236
	5	0.1	0.0	0.4	420	0	1826
	4	0.1	0.0	0.2	311	0	1209
15	8	0.1	0.0	0.5	140	0	123
	7	0.8	0.0	5.43	1912	0	15764
	6	0.3	0.0	0.7	619	0	2221
	5	4.4	0.0	26.8	13430	0	89648
	4	2.5	0.1	15.0	8503	158	53217
20	8	0.6	0.1	2.0	716	0	3186
	7	9.3	0.0	57.2	15431	0	105959
	6	3.2	0.2	12.8	5229	318	21918
	5	119.5	5.1	774.5	218267	7162	1287754
	4	141.8	3.0	810.5	299105	6970	1687972
30	8	56.4	0.9	600.9	39658	0	449931
	7	12321.0	49.8	244742.5	11242258	11908	230685549
	6	3082.0	4.0	96398.0	2451303	1653	75321931

Table 3.2 : Branching Rule 2 (depth first search)

Note that for the largest value of $|V|$, Tables 3.1 and 3.2 do not include results for D = 4 or 5. For these values of D we were unable to completely solve all 50 test problems in a reasonable time. An alternative algorithm is developed for these D in the next section. Branching Rule 1, despite its simplicity and intuitive appeal, does not perform particularly well. This inferior performance is due in part, to poor upper bounds. Branching Rule 2 generates diameter feasible solutions at each node in the search tree and thus tends to maintain a tighter upper bound. However, neither method is suitable as the diameter bound is tightened. This led to an improved algorithm (Branching Rule 3 in [2]) for the case D = 4. This is generalized in the next section.

To determine the value of the Heuristic we consider the statistic (H-O)/O, where

H : the objective function value obtained by the heuristic,

and

O : the optimal objective function value.

The relevant statistical information is summarized in Table 3.3. Note that the last column of this table gives the percentage of problems for which the heuristic yields a solution within 10% of the optimum.

$\|V\|$	D	STATISTIC X = (H-O)/O					
		Mean	Median	Std. Dev.	Min	Max	% with X ≤ .10
10	8	.0174	0	.0717	0	.4544	94
	7	.1158	.1120	.0747	.0011	.3231	40
	6	.0537	0	.1240	0	.5776	84
	5	.1711	.1352	.1701	.0017	.8482	38
	4	.0986	.0055	.1674	0	.7126	66
15	8	.0164	0	.0289	0	.1224	98
	7	.0971	.0833	.0825	.0029	.3853	60
	6	.0408	.0130	.0664	0	.3037	88
	5	.1640	.1167	.1374	.0129	.5852	38
	4	.1089	.0551	.1277	0	.5081	62
20	8	.0290	.0011	.0493	0	.1845	86
	7	.1032	.0831	.0792	.0061	.4163	62
	6	.0995	.0576	.1215	0	.4941	58
	5	.1915	.1453	.1378	.0191	.5474	32
	4	.1327	.0755	.1891	0	1.0214	60
30	8	.0681	.0136	.0964	0	.3329	76
	7	.1486	.0974	.1479	.0059	.8543	52
	6	.1340	.1047	.1503	0	.8854	48
	5	.2196	.1892	.1623	.0098	.7688	26
	4	.1906	.1202	.1986	0	.8531	44

Table 3.3 : Performance of Heuristic

4. NEW SOLUTION PROCEDURES

The method developed in this section involves the decomposition of the given MWST-D problem into a number of simpler subproblems which involve the determination of minimum weight spanning trees having a fixed root vertex and bounded height. Note that the height of a rooted tree is taken to be the distance of the vertex furthest from the root.

We call a tree T with root r and maximum height h a **(r,h)-tree**. The subproblem that arises in our decomposition is the following :

> **Problem 4.1** : Given a weighted graph G with a distinguished root vertex r and a positive integer h, find a minimum weight (r,h)-spanning tree in G.

We refer to this problem as the **MWST-(r,h) problem.**

The MWST-D problem can be solved by considering a number of MWST-(r,h) problems. For the case D even it is sufficient to consider $|V|$ MWST-$(r,\frac{1}{2}D)$ problems. For the case D odd we consider $|E|$ MWST-$(r,\lfloor \frac{1}{2}D \rfloor)$ problems where r is formed by contracting an edge. Consequently we focus our attention on Problem 4.1.

We can formulate the MWST-(r,h) problem as a MILP problem. Our formulation involves transforming the given weighted graph $G = [V,E]$ with root vertex r into a weighted directed graph $G' = [V,\mathcal{A}]$ as follows. Let $V\backslash\{r\} = \{1,2,\ldots,n-1\}$. Then

$$\mathcal{A} = \{(r,j) : j = 1,2,\ldots,n-1\} \cup \{(i,j) : 1 \le i \ne j \le n-1\}.$$

The weight w_{ij} of the arc (i,j) is taken to be the weight of the corresponding edges (i,j) in G; if the edge is not in G then the corresponding weight is ∞. The MILP formulation is :

$$\text{Minimize} \sum_{(i,j)\in\mathcal{A}} w_{ij}\, x_{ij} \tag{4.1}$$

subject to

$$x_{rj} + \sum_{\substack{i=1\\ i\neq j}}^{n-1} x_{ij} = 1, \qquad \text{for } j = 1,2,\dots,n-1, \tag{4.2}$$

$$x_{ij} = 0 \text{ or } 1, \qquad \text{for } (i,j) \in \mathcal{A} \tag{4.3}$$

and for all ordered h-subsets $\{i_1, i_2, \dots, i_h\}$ of $\{1,2,\dots,n-1\}$

$$\sum_{t=1}^{h-1} x_{i_t i_{t+1}} - x_{ri_1}, \le h - 2 \tag{4.4}$$

and

$$\sum_{j=1}^{t-1} x_{i_j i_{j+1}} + x_{i_t i_1} \le t - 1, \qquad \text{for } t = 2,3,\dots,h-1 \tag{4.5}$$

That the above formulation does indeed solve Problem 4.1, is easily established as follows. Consider a solution (x_{ij}) of constraints (4.2) to (4.5). Restriction to the arcs of G′ with $x_{ij} = 1$ gives rise to a directed graph G* having the following properties. Each vertex of $V\backslash\{r\}$ has, by restriction (4.2), indegree exactly one. Constraint (4.5) ensures that G* has no cycles of length at most h. Further, constraints (4.2) and (4.4) together ensure that G* has no cycles of length greater than h. Thus G* consists of directed paths from the root vertex r having (by (4.4)) length at most h. Consequently the MILP formulation (4.1) to (4.5) solves the MWST-(r,h) problem.

The combinatorial nature of constraints (4.4) and (4.5) restricts the usefulness of the above MILP formulation for solving the MWST-D problem. However, for small values of h, good results are obtained as detailed in Tables 4.1 and 4.2. As in Section 3 our computational results were obtained on a SUN SPARC Workstation using 50 test problems for each order.

$\lvert V\rvert$	CPU Ave	TIME Min	(sec) Max	No. of Subproblems Ave	Min	Max
20	5.7	4.8	6.5	0.0	0	2
30	31.9	28.5	35.6	0.0	0	0
40	113.0	102.0	120.3	0.1	0	2
50	366.0	318.8	404.1	0.2	0	2
100	7342.5	6928.3	8050.6	0.2	0	3

Table 4.1 : MWST-4 Results

$\|V\|$	CPU TIME (sec) Ave	Min	Max	No. of Subproblems Ave	Min	Max
10	2.6	1.4	7.9	7.6	0	102
15	32.2	18.3	147.3	41.9	0	503
20	204.9	93.7	681.1	155.4	2	787

Table 4.2 : MWST-6 Results

Given the limited application of the MILP formulation (4.1) to (4.5) we now develop procedures exploiting the structure of a (r,h)-tree which cater for the solution of problems with larger D.

Consider Problem 4.1. We call a spanning tree T of G **feasible** if it is a (r,h)-tree. Observe that the vertices of a feasible T can be partitioned into layers $L_0 = \{r\}$, $L_1,\ldots,L_h$, where L_i consists of the vertices at distance i from the root vertex r. In the previous methods mentioned in Section 3 for solving the MWST-D problem, branching was defined by inclusion and exclusion of edges. Here we consider branching on vertices rather than edges.

The relaxed subproblem is defined by a partial partitioning of the vertices of G into layers $L_0 = \{r\}$, $L_1,\ldots,L_h$. We denote by U those vertices of G which are not in any L_i, $i = 0,1,\ldots,h$. Thus, $L_0,L_1,\ldots,L_h,U$ is a complete partition of V(G). Observe that when $U = \phi$, the layer structure restricts the choice of edges to only those connecting vertices in adjacent layers. However, solving the MWST problem on the corresponding graph does not necessarily yield a (r,h)-tree. For example, the MWST solution does not require that every vertex in L_1 be joined to the root. Choosing the edge of smallest weight from a vertex in L_i to a vertex in L_{i-1} will resolve this problem. However, this does not extend to the case when $U \neq \phi$. This problem can be resolved by considering a directed version which we now describe.

Problem 4.2 : Given a weighted directed graph G′ with a distinguished root vertex r, find a minimum weight spanning tree in G′ in which there is a directed path from r to every other vertex of G′.

We refer to this problem as the **DMWST(r) problem**. This problem was first considered by Edmonds [4] and the best solution procedure to date, due to Gabow et. al. [5], has complexity $O(|V|^2)$.

Observe that given the MWST-(r,h) problem with a complete vertex partition $L_0 = \{r\}, L_1, \ldots, L_h$, we can form a directed graph G′ by orienting the edges between adjacent layers from L_{i-1} to L_i, $i=1,2,\ldots,h$. A solution of the DMWST(r) problem in G′ is clearly a solution of the MWST-(r,h) problem. Our strategy is to work towards a complete vertex partition by branching on vertices not yet assigned to levels. We make use of the algorithm for solving Problem 4.2 at each step. We now describe our procedure in detail.

Consider Problem 4.1. We introduce some notation to assist in the description of our subproblems and branching rules. Let $\Pi = \{L_0, L_1, \ldots, L_h, U\}$ denote a complete vertex partition of G with $L_0 = \{r\}$. We begin with the partition having $L_i = \phi$ for $i = 1,2,\ldots,h$, and $U = V(G)\backslash\{r\}$ and at each stage the partition is modified by transferring a vertex out of U. At each stage $L_0, L_1, \ldots, L_h$ represents a partial layer structure and this structure restricts some of the edges of our original graph G. Thus with each subproblem P we have an associated graph G_p and partition Π_p. For simplicity we write $\Pi_p = \{r, L_1, \ldots, L_h, U\}$.

With the graph G_p we associate a directed graph G'_p. As G'_p is the graph we work with we now describe its structure by specifying its arc set. Every vertex in L_i, i =0,1,...,h-1, is joined by a directed edge to every vertex in $L_{i+1} \cup U$. Every vertex in U is joined by a directed edge to every vertex in L_i, i=2,3,...,h. Further, every pair of vertices in U is joined by a pair of oppositely directed edges. The weights on all directed edges are taken to be the weights of the corresponding edges in G. Figure 4.1 shows the basic structure of G'_p.

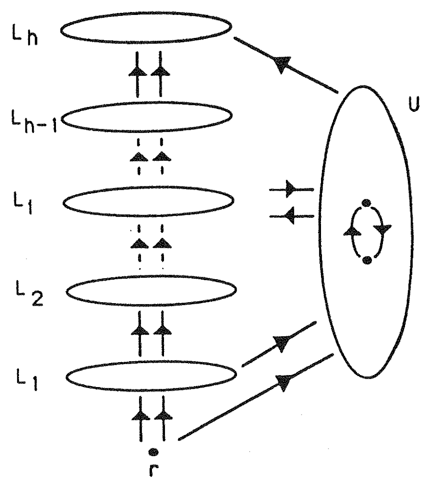

Figure 4.1 : The directed graph G'_p

Let T'_p be a solution of the DMWST(r) problem on G'_p. If the height of T'_p is at most h then we have an optimal solution to the subproblem P. If this is not the case we find the set W of end vertices of T'_p that are joined to the root r by a path of length greater than h. We then find the set of vertices $U' \subseteq U$ defined as :

$$U' = \{u \in U : u \text{ is on some } (r,w)\text{-path}, \ w \in W\}.$$

Note that $U' \neq \phi$ since T'_p has height greater than h. Now we create subproblems by finding the vertex $x \in U'$ having maximum degree in T'_p and branching according to the following rule :

Branching Rule 4.1 : Given x we create h subproblems $P_1, P_2, \ldots, P_h$ from P by modifying Π_p as follows :

$$\Pi_{p_i} = (r, \ L_1, \ldots, L_{i-1}, L_i \cup \{x\}, \ L_{i+1}, \ldots, L_h, U \setminus \{x\}) \ ,$$

$$i = 1, 2, \ldots, h.$$

A vertex of maximum degree is chosen in an effort to obtain the maximum change in the subproblems generated.

We implemented (using depth first search) the above branching rule using the same hardware as used previously. The detailed algorithm is presented in the form of a flow diagram in Figure 4.2. In our description we use the following notation. In the case of even D, for every vertex $i \in V$ we consider the MWST - (i,h) problem. The corresponding initial vertex partition is denoted by

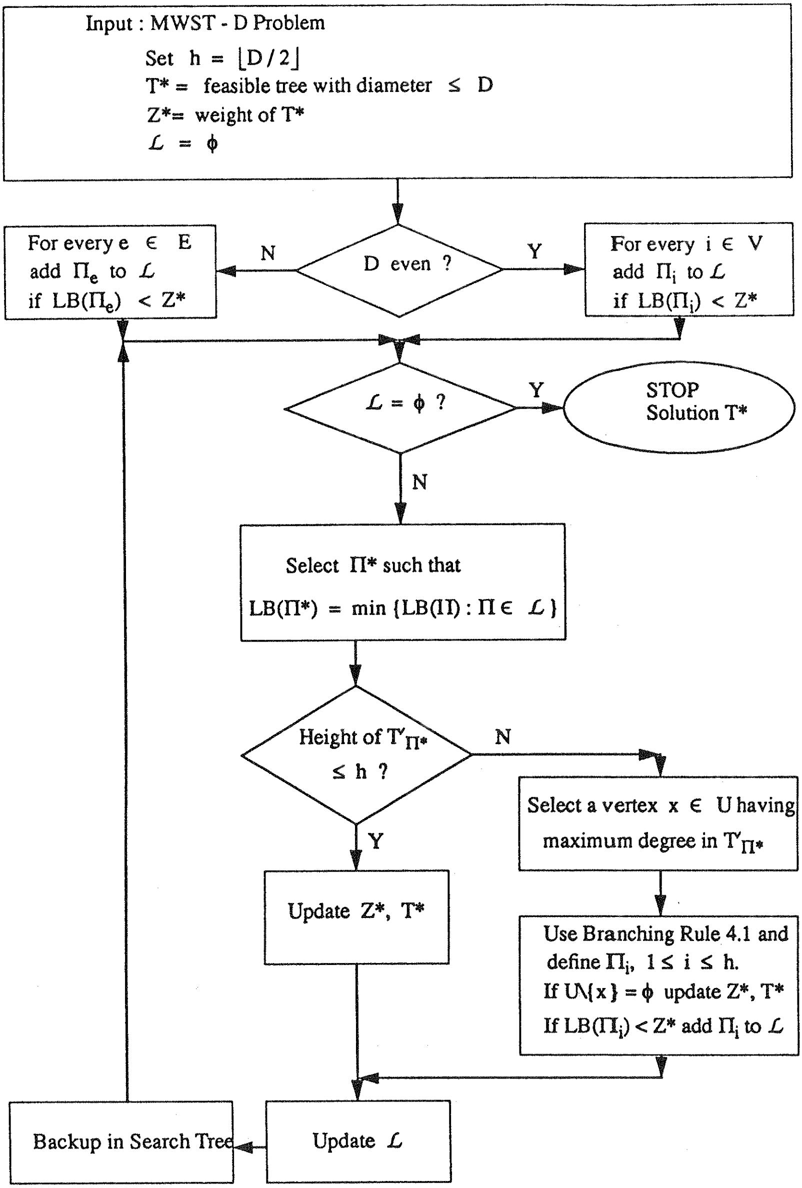

Figure 4.2 : Flowchart of Branch and Bound Algorithm.

$\Pi_i = (L_0, L_1, ..., L_h, U)$, where $L_0 = \{i\}$, $L_j = \phi$ for $1 \le j \le h$ and $U = V \backslash L_0$. Similarly, for odd D, $\Pi_e = (L_0, L_1, ..., L_h, U)$ denotes the initial partition for the MWST-(r_e, h) problem, where r_e is the vertex obtained by contracting the edge e. Note that here $L_0 = \{r_e\}$, $L_j = \phi$ for $1 \le j \le h$ and $U = V(G \cdot e) \backslash L_0$, where $G \cdot e$ denotes the graph obtained by contracting the edge e of G. The weight of the tree T'_{Π}, the optimal solution of the DMWST problem, is denoted by LB(Π). Table 4.3 gives the computational results for the set of test problems described in the previous section.

|V|	D	CPU TIME (sec) Ave	Min	Max	No. of Subproblems Ave	Min	Max
10	8	0.0	0.0	0.6	101	0	1778
	7	0.0	0.0	0.2	260	0	1832
	6	0.0	0.0	0.1	446	0	1933
	5	0.0	0.0	0.1	1347	0	1991
	4	0.0	0.0	0.1	1972	0	2276
15	8	0.5	0.0	3.4	1098	0	5717
	7	0.5	0.0	1.6	1427	0	3834
	6	0.4	0.0	1.2	1762	0	3226
	5	0.6	0.0	1.3	2171	0	3516
	4	0.3	0.1	1.6	2386	2080	3912
20	8	5.7	0.0	41.6	4789	0	29550
	7	3.1	0.0	7.2	2578	0	5460
	6	3.1	0.4	12.0	3855	1746	11097
	5	3.5	2.0	7.1	3616	2370	6313
	4	1.4	0.4	3.6	3511	2420	5786
30	8	1896.1	0.0	37396.0	637490	0	12839037
	7	179.3	14.1	1872.3	64317	2843	710092
	6	507.2	3.5	5164.8	210009	2175	2172384
	5	64.1	20.6	190.7	18467	5679	55627
	4	28.7	7.2	117.3	19802	6032	77338
40	4	491.8	28.6	1995.6	171950	10760	695796
50	4	7375.7	1328.9	29073.5	1596730	272770	6252282

Table 4.3 : Results for Branching Rule 4.1.

Our implementation makes use of the heuristic procedure described in Section 3 for computing an upper bound. Further, by considering each vertex as the root, application of the Heuristic $|V|$ times produces an ordering of the vertices that we use as the root vertex of the layered structure tree. The ordering is according to the weight of the Heuristic solution from each root vertex.

From the results given in Tables 4.1 to 4.3 we observe that for large $|V|$, the MILP formulation (4.1) to (4.5) significantly outperforms Branching Rule 4.1 for the case $D = 4$. However, for $D \geq 6$ this situation is reversed. Further, in comparison with the tables given in Section 3, we note that the new branching rule is superior for the case of tighter diameter restrictions and the range of superiority increases with $|V|$. This is, of course, consistent with the manner in which the rules were devised.

ACKNOWLEDGEMENTS

This work has been supported by ARC Grant A48932119.

REFERENCES

1. N.R. Achuthan and L. Caccetta, **"Minimum weight spanning trees with bounded diameter"**, Australasian Journal of Combinatorics, Vol. 5, 1992, 261-276. (Addendum: Vol 8, 1993, 279-281).

2. N.R. Achuthan, L. Caccetta, P. Caccetta and J. Geelen, **"Algorithms for the minimum weight spanning tree with bounded diameter problem"**, in Optimization Techniques and Applications, Vol. 1 (P.H. Phua et.al. Editor), World Scientific (1992), 297-304.

3. L. Caccetta, **"Graph theory in network design and analysis"**, in Recent Studies in Graph Theory (V. Kulli, editor). 1989, pp. 29-63.

4. J. Edmonds, **"Optimum branchings"**, J. of Research of the National Bureau of Standards (Math. and Math. Phys.), 71B (1967), 233-240.

5. H.N. Gabow, Z. Galil, T. Spencer and R.E. Tarjan, **"Efficient algorithms for finding minimum spanning trees in undirected and directed graphs"**, Combinatorica, Vol. 2, 1986, pp.109-122.

6. M.R. Garey and D.S. Johnson, **"Computers and Intractability - A guide to the Theory of NP-completeness"**, Freeman, San Francisco, 1979.

7. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, **"The Travelling Salesman Problem"**, John Wiley & Sons, 1985.

8. C.H. Papadimitrou and M. Yannakakis, **"The complexity of restricted spanning tree problems"**, JACM, Vol. 29, 1982, pp. 285-309.

(Received 19/8/93)