*2009 International Conference on Machine Learning and Computing*
*IPCSIT vol.3 (2011) © (2011) IACSIT Press, Singapore*

# An Improved Node Partitioning Algorithm for the CMST Problem

Jun Han [1 +], Ming Lei [1], Dianfu Ma [1], Xiang Long [1], Yaling Huang [2]

[1] School of Computer Science and Engineering, Beihang University, China
[2] School of Automation Science and Electrical Engineering, Beihang University, China

**Abstract.** The capacitated minimum spanning tree (CMST) problem is one of the most fundamental and significant problems in the optimal design of communication networks. In this paper several methods are proposed to improve existing node-oriented branch and bound algorithm for the CMST problem. Techniques for acquiring tighter lower bound are emphasized, while regulations for faster traversing of the search tree are proposed on the basis of comprehensive observation of the search procedure. Computational experiences demonstrate that the proposed methods significantly improved the efficiency of the existing algorithms. Moreover, as an exact algorithm, the proposed one is the first algorithm that solved the 41 node te-class benchmark problem instances.

**Keywords:** Branch and bound, Search tree, Pruning, Penalty

## 1. Introduction

### 1.1. The CMST problem

In the capacitated minimum spanning tree problem, the objective is to find a minimum cost tree spanning a given set of nodes such that some capacity constraints are observed. Considering a connected graph $G=(V,A,b,c)$ with node set $v=\{0,1,....,n\}$ and arc set $A$, node $0$ is a special node called the center node and is the root of the tree. Each node $i>0$ in $V$ has a unit flow demand from the center node, while a non-negative arc weight $c_{ij}$ represents the cost of using arc $(i,j)$ in a solution tree. A rooted sub-tree (or component) $r$ of a tree spanning $V$ is defined as its maximal sub-graph that is connected to the center by arc $(0,i)$, which is called central arc. To satisfy the capacity constraint, the flow on each arc must not exceed a given capacity $K$. The capacitated minimum spanning tree problem is NP-hard when $2<k<(n/2)$, as proved by Papadimitriou in [14]. The CMST problem is one of the most fundamental problems in the optimal design of communication networks, as well as in general network optimizations. A significant amount of research has been done in designing heuristics for solving CMST problem [1][2][3][7][8][9][10].

Regarding exact methods, which guarantee finding the optimal solutions, Chandy and Rusell[4] proposed a branch and bound algorithm, in which the sub-problem is branched by established or not the first not yet established arc, counted from the center, of an infeasible sub-tree, and then modify the cost matrix according to the additional constraints. With this algorithm, problems with up to 40 nodes are considered. Chandy and Lu[5] branch sub-problems by requiring two nodes of sub-tree exceeding the capacity to be in the same sub-tree or not. Problems with up to 50 nodes are considered. Kershenbaum and Boorstyn[12] proposed a branch and bound algorithm where a sub-problem is branched by including or excluding a node form a specific sub-tree and computation results for 20-node problems are given. Gouveiaand Paixao[10] proposed a dynamic programming approach and the algorithm is applicable to instances with up to approximately a dozen nodes. Malik and Yu[13] presents another branch and bound algorithm with Lagrangean subgradient optimization. Computation results with up to 50 nodes are reported.

+ Corresponding author. Tel.: +8601082316340; fax: +8601082316340.
*E-mail address*: jun_han@buaa.edu.cn

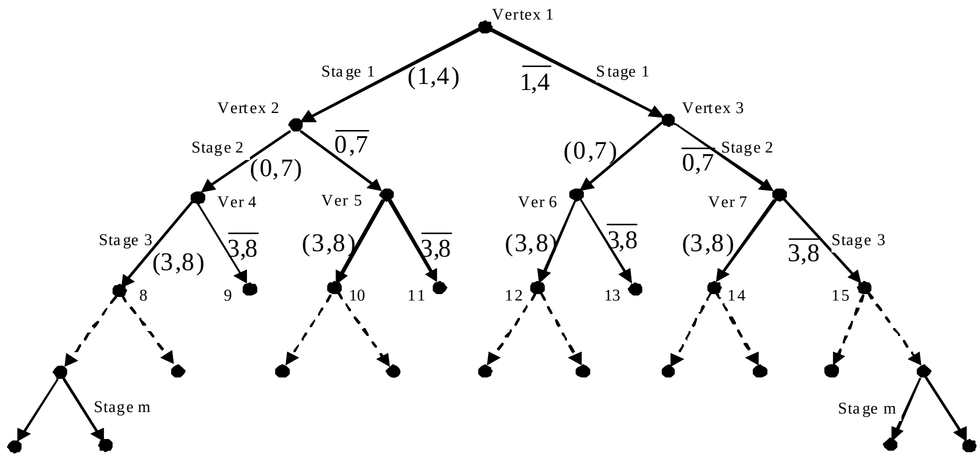


Fig. 1. The Search Tree

Han et al's arc-oriented branch and bound algorithm [11] was the first exact algorithm that solved part of the benchmark problems in OR-Library and solved 100 node random generated problem instances, and the algorithm was completely dominated later by a node-oriented branch and bound algorithm [15]. As the most recent achievement, more efficient node-oriented branch and bound algorithm was proposed by Han et al. [16], in which techniques for fast traversing of the search tree were developed and a heuristic method was proposed and employed to generate better initial solution.

From section 2, an improved node-oriented branch and bound algorithm will be introduced, and computational results as well as comparisons with the existing best exact algorithm [16] will be presented.

## 1.2. The Node Partitioning Algorithm

### The Search Tree

An m-stage binary search tree is defined for a problem with n nodes, where m equals the number of pairs of nodes $\binom{n}{2}$, as shown in Figure 1. The branching within each stage corresponds to the decision of either forcing a certain pair of nodes to be in the same rooted sub-tree (component) or not. It is also defined that traveling along the left branch represents for grouping a pair of nodes and traveling along right branch represents for splitting a pair of nodes. In Figure 1, for example, at stages 1 and 2, traveling from Vertex 1 to Vertex 2 then to Vertex 5 represents grouping nodes 1 and 4 (denoted as $(1,4)$), but never grouping nodes 0 and 7(denoted as $\overline{0,7}$).

Depth-first Search is used when traversing the search tree, and backtracking is performed when any of the following situations occurs: a. Current partitioning makes the solution infeasible; b. A feasible solution has been found; c. Total cost of current partial solution is already more expensive than the cost of an existing solution.

### Pruning the Searching Tree

Techniques for fast traversing and pruning of the search tree are needed to improve the efficiency of the search procedure. First, all the candidate arcs are sorted in the order of ascending arc cost and each of them is given an arc number, from 1 to m, corresponding to the m stages of the search tree from top to bottom. The branching within stage i corresponds to the decision of whether to group the two extremities of arc i. This arrangement brings many features that lead to faster pruning of the search tree, as discussed in [15] and [16].

At each vertex in the search tree, a modified Kruskal's algorithm is used to calculate the lower bound at this vertex. The MST for the whole problem is completed with the capacity constraint relaxed. A group is called an "original group" at a vertex, if it is formed before the procedure starts constructing the MST for the whole problem at this vertex. We name this spanning tree PPMST (Partially Partitioned Minimum Spanning Tree). When constructing the PPMST, the following rules are respected: 1. Do not make cycles; 2. Do not merge alien groups.

Usually, a PPMST is an infeasible solution, whose cost is a lower bound on solutions at this vertex. If thislower bound is greater than the global upper bound, the sub-tree rooted at this vertex in the search tree is pruned.

To acquire tighter lower bound, a penalty cost is added to the PPMST bound [15][16]. The calculation of the penalty cost in previous algorithms will be briefed in the next section to help immediate introduction of the proposed technique of achieving higher penalty value.

## 2. The Improved Algorithm

### 2.1. Modification of the backtrack regulation

It is observed that, in the search tree, after traveling backward to a vertex along its left branch, go forward along its right branch is not always necessary, we could go backward to its parent vertex if the arc corresponds to the stage is a central arc and the backtracking is not caused by an infeasibility of current partitioning.

Suppose stage 2 of the search tree in Figure 1 corresponds to the pair of end nodes of a central arc $(0,7)$ (Node 0 is the center node). If a feasible solution is found or the total cost of current partial solution is already more expensive than that of an existing solution at vertex 4, the procedure backtracks to vertex 2 and then to vertex 5 in the previous algorithms [15][16]. Suppose the tree solution $A$ shown in figure 2 is the best feasible solution in the search space below vertex 5.

It is noticed that, in the search tree shown in figure 1, the current solution must be feasible and node 7 is connected to the center node by arc $(0,7)$ when backtracking to vertex 4. But when backtracking to vertex 2 and then to vertex5, the group in which node 7 exists is not connected to the center node. Because $A$ is found in the search space below vertex 5, there must exist at least one arc, the corresponding stage of which is below that of arc $(0,7)$, in the path from node 7 to the center node in $A$ .Suppose arc $(3,7)$, as shown in Figure 2, is an arc whose corresponding stage is below stage 2 and therefore the weight of arc $(3,7)$ is no less than the arc $(0,7)$. Now, we connect node 7 to the central node directly and eliminate the arc $(3,7)$, we get the new tree solution $A'$ as shown in figure 3.

The weight of arc $(0,7)$ is no more than that of arc $(3,7)$. Consequently the cost of the new tree solution can not exceed the original one. And it must be feasible because it just divide one group into two which would not lead to form a group whose size exceed the given capacity. It is guaranteed that the new tree solution $A'$ can be found in the search space along the left branch of vertex 2 because the new tree solution contains arc $(0,7)$. As a result, it can be concluded that we cannot find a better feasible solution than the current one in the search space below vertex 5.The modified regulation accelerates the pruning of the search tree and improves the efficiency of the algorithm by saving unnecessary search.

### 2.2. Improvement of the Additional Penalty

To acquire a tighter lower bound, an additional penalty is added to the PPMST bound. At each vertex in the search tree, the PPMST is generally infeasible due to excessive size of rooted sub-tree and thereafter too much flow on central arcs. This implies that more sub-trees need to be connected to the center directly to get a feasible solution. It is noted that for an N-node problem, if the capacity constraint is K, then no more than K nodes can be in one rooted sub-tree in any feasible solution. Therefore, there has to be at least S rooted sub-trees in a feasible solution, where S equals $\lceil N-1/K \rceil$. If there are M rooted sub-trees in the infeasible PPMST, then at least L=S-M more original groups must be connected to the center directly. We now define

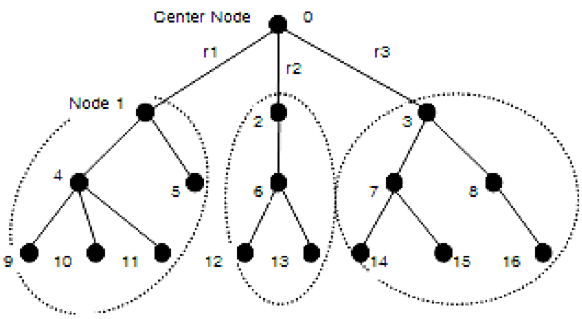

Fig. 2. Tree Solution $A$

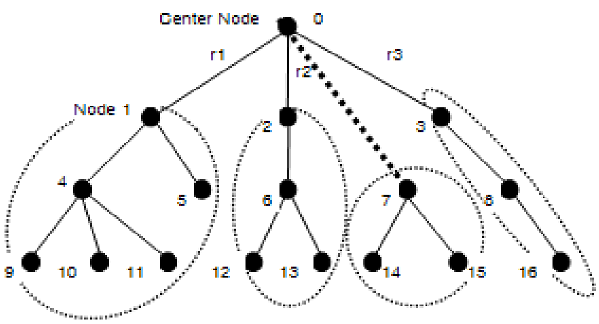

Fig. 3. Tree Solution $A'$

$C_p{}'$ to be the least cost increase to connect L further original groups directly to the center. Our efforts will focus on finding a "penalty cost" $C_p$, which is less than $C_p{}'$, but as greater as possible.

**The previous method to calculate the penalty cost**

The calculation of the penalty cost in [15] and [16] is as follows. Suppose there are $k_m$ original groups in $C_m$ (component $m$), where $m = 1,2,..,M$ .Let $CC_m[i]$ be the sequence (ascending order) of costs for connecting the original groups (excluding the one already connected to the center) in component $m$ directly to the center, $CE_m[i]$ be the sequence (descending order) of the costs of arcs linking all original groups in $C_m$, where $i = 1,2,..,k_m$. Define $CP_m[i] = CC_m[i] - CE_m[i]$, $i = 1,2,..,k_m$, then the least cost increase for joining $l$ original groups in $C_m$ directly to the center, must be greater than or equal to the sum of the first $l$ elements in sequence $CP_m[i]$. We merge the sequences $CP_m$, $m = 1,2,..,M$, into one sequence $CP[i]$, where $i = 1,2,..,\sum_{m=1}^{M} k_m$, then $C_p{}'$, which is the least cost increase for joining L original groups in the whole PPMST directly to the center, must be greater than or equal to the sum of the first L elements in $CP[i]$. The penalty cost $C_p$ is defined to be sum of the first L elements in $CP[i]$. And it is added to the cost of PPMST to get a much tighter lower bound.

Suppose there is a 21-node problem with a capacity constraint 4. At a certain vertex in the search tree, the PPMST is constructed by linking the original groups $OG_1$ to $OG_9$, as shown in Figure 4. Since S $= \lceil N-1/K \rceil = \lceil 20/4 \rceil$=5, we know there has to be at least L more components constructed to acquire a feasible solution, where L=S-M=5-3=2. This means at least 2 more original groups have to be connected to the center node directly. According to the method introduced above we will get $CP = [1,4,7,7,8,14]$ eventually. Since at least 2 more original groups need to be connected to the center directly to acquire a feasible solution, we get the penalty cost $C_p = CP[1] + CP[2] = 5$.

**Proposed Method for Calculating the Penalty Cost**

The above method simply subtracts the arc weight of one sequence from another within a component. The lack of pertinence leads to relatively small, sometimes even negative penalty value.

It is noticed that a circle will form when an original group in the PPMST is connected to the centre node directly. To guarantee that the PPMST is a tree and the number of subtrees is increased, an arc which connects two original groups in the circle must be eliminated. In the PPMST shown in figure 4, when $OG_4$ is connected to the center node directly, the arc which connects $OG_3$ and $OG_4$ must be eliminated.

Based on the above fact, a new method of calculating the penalty cost is proposed. Suppose there are $k$ original groups in the PPMST (excluding the one already connected to the center). For any original group $OG_i$, let $CC_i$ be the least cost for connecting the original group directly to the center, $CE_i$ be the cost of the weightiest arc linking two original groups in the circle forms after $OG_i$ is connected to the centre node, where

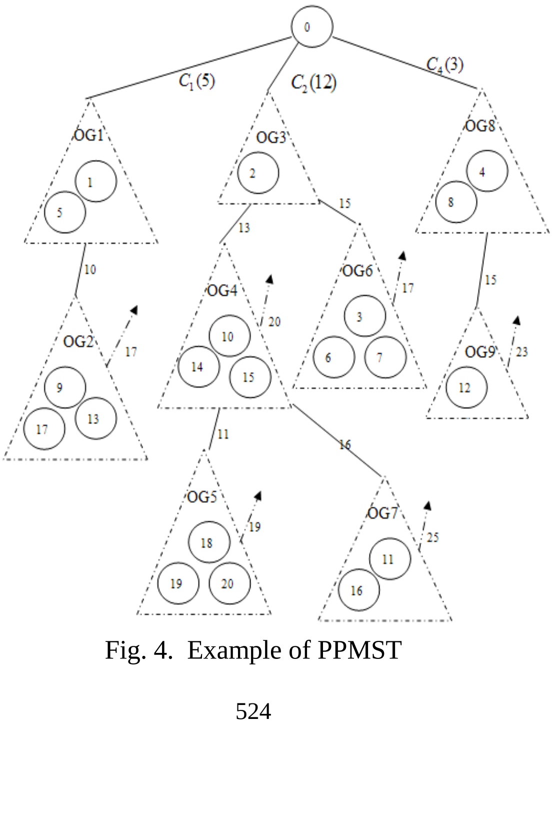

Fig. 4. Example of PPMST

$i = 1,2,..,k$ , then the least cost increase for joining the original group $OG_i$ directly to the center is $CC_i - CE_i$ ; We define $CP_i = CC_i - CE_i$ , $i = 1,2,..,k$ , then we put all the $CP_i$ into the sequence $CP[]$ (ascending order).

We define the new penalty cost $C_p''$ to be sum of the first L elements in $CP'[]$. Then, $C_p''$ is no more than $C_p'$ which is the least cost increase for joining L original groups in the whole PPMST directly to the center.

Regarding the example in figure 4, the process of calculating the penalty cost according to our new method is as follows. The least cost for joining the original group $OG_2$ directly to the center $CC_2 = 17$, the cost of the weightiest arc linking two original groups in the circle forms after $OG_2$ is connected to the centre node $CE_2 = 10$, then, the least cost increase $CP_2 = CC_2 - CE_2 = 17 - 10 = 7$. Similarly, we get $CP_4 = 7;\ CP_5 = 6;\ CP_6 = 2;\ CP_7 = 9;\ CP_9 = 8$. Then we have $CP' = [2,6,7,7,8,9]$ . Since at least 2 more original groups need to be connected to the center directly, we get the new penalty cost $C_p'' = CP'[1] + CP'[2] = 2 + 6 = 8 > 5 = C_p$ .

The above discussion shows that, compared to the original one, our new method could get higher penalty cost and therefore tighter lower bound with which the pruning procedure will be accelerated and the running time of the algorithm will be decreased.

## 3. Computational Experience

OR-Library (http://people.brunel.ac.uk/~mastjjb/jeb/info.html) is a collection of test data sets for a variety of operations research problems. Most modern researchers studying the CMST problem show their achievements by presenting computational test results for solving the benchmark problems in OR-Library.

As introduced in Section 1, Han et al's node-oriented branch and bound algorithm is the latest achievement on exact algorithm for CMST problem [16]. It is also the only work reported to have solved up to 100 node problem instances and also the only one that has solved part of the benchmark problems in OR-Library. In our proposed algorithm, the same neighborhood search algorithm is employed for generating the initial solution as in [16].

Table 1 is a comparison of the proposed algorithm and the previous branch and bound algorithm [16] for solving the "tc" class 41 node benchmark CMST problems provided by OR-Library. Table 1 shows that, compared with the previous method [16], the proposed algorithm reduces the searching steps by 88.4%, and saves computation time by 81.7% on an average. Table 2 is the computational results of the proposed algorithm for solving the "te" class 41 node benchmark problems provided by OR-Library. It is the first time that these benchmark problems are solved by an exact algorithm.

Table 1. Comparison of the Proposed Method and the Existing Method

| Problem File Name (k=10) | Existing Method | | Proposed Method | | Percentage of Reduced Searching Steps | Percentage of Reduced Running Time |
|---|---|---|---|---|---|---|
| | Searching Steps | Running Time (Seconds) | Searching Steps | Running Time (Seconds) | | |
| tc41-1 | 5,150,790 | 11.5 | 619,365 | 1.1 | 88.0% | 90.8% |
| tc41-2 | 274,788,265 | 516.2 | 27,508,112 | 58.6 | 90.0% | 88.7% |
| tc41-3 | 184,843,111 | 574.4 | 20,861,295 | 49.6 | 88.7% | 91.4% |
| tc41-4 | 16,261,121 | 26.5 | 2,313,718 | 10.2 | 85.8% | 61.4% |
| tc41-5 | 64,559,517 | 134.1 | 6,807,023 | 31.7 | 89.5% | 76.3% |
| Average | | | | | 88.4% | 81.7% |

Table 2 Results of the Proposed Method for Solving Te-41 Problems

| Problem File Name (k=10) | Optimal Solution | Searching Steps | Running Time(Seconds) |
|---|---|---|---|
| te41-1.txt | 596 | 3,857,896,649 | 127,465 |
| te41-2.txt | 573 | 293,718,395 | 15,815 |
| te41-3.txt | 568 | 44,641,209,715 | 1,379,947 |
| te41-4.txt | 596 | 1,252,748,059 | 51,403 |
| te41-5.txt | 572 | 463,858,733 | 24,242 |

The computations were performed on a machine with Pentium IV 3.39 GHz CPU and 2 GB RAM, and the application programs were run on Windows XP operating system. All the test problems and respective solutions found by our algorithm can be accessed at: http://act.buaa.edu.cn/hanjun/Tutor_HanJun_En_2.htm

# 4. Conclusion

We have proposed an improved node-oriented branch and bound algorithm for the Capacitated Minimum Spanning Tree problem and have presented the comparison between our improved algorithm and the previous ones. New methods for acquiring higher penalty value to the lower bound, as well as new regulations for traversing the search tree and for checking the feasibility are proposed. Computational experience shows that the proposed algorithm completely dominates existing exact algorithms.

# 5. Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC grant number: 60842008 and 90818028) and China 863 High-tech Programme (Project No. 2007AA010301). The authors would also like to thank all those who took part in the focus groups and experiments.

# 6. References (This is “Header 1” style)

[1] Ahuja, R.K., J. B. Orlin and D. Sharma “A composite very large-scale neighborhood structure for the capacitated minimum spanning tree problem”. *Operations Research Letters*, 2003, vol.31, pp.185-194.

[2] Ahuja, R.K., Orlin, J.B. and Sharma, D.. “Multi exchange neighborhood search algorithms for the capacitated minimum spanning tree problem”. *Mathematical Programming*, 2001,vol.91,pp. 9-39.

[3] Altinkemer, K. and Gavish, B. “Heuristics with constant error guarantees for the design of tree networks”. *Management Science*, 1988,vol.32, pp. 331 - 341.

[4] Chandy, K.M. and Russell, R.A. “The design of multipoint linkages in a teleprocessing tree network”. *IEEE Transactions on Computers*,1972, vol.21, pp.1062-1066.

[5] Chandy, K.M. and Lo, T. “The capacitated minimum spanning tree”. *Networks*,1973, vol.3,pp.173-181.

[6] Gouveia, L. and Martins, P. “A hierarchy of hop-indexed models for the capacitated minimum spanning tree problem”. *Networks* ,2000,vol.35, pp.1-16.

[7] Gouveia, L. and Martins, P.“The capacitated minimum spanning tree problem: revisiting hop-indexed formulations”, *Computers & Operations Research*,2005, vol.32, pp.2435-2452.

[8] Martins, P. “Enhanced second order algorithm applied to the capacitated minimum spanning tree problem”, *COMPUT OPER RES*, 2007,vol.34, pp.2495-2519.

[9] Marc Reimann, Marco Laumanns, “Savings based ant colony optimization for the capacitated minimum spanning tree problem”. *Computers and Operations Research*, 2006, vol.33, no.6, pp.1794-1822.

[10] Gouveia, L. and Paixao, J. “Dynamic programming based heuristics for the topological design of local access networks”. *Annals of Operations Research*,1991,vol.33, pp.305-327.

[11] Han, J., McMahon, G.B. and Sugden, S.J. “A branch and bound algorithm for capacitated minimum spanning tree problem”. *Lecture Notes in computer Science*,2002 vol.2400,pp.404-408.

[12] Kershenbaum, A. and Boorstyn, P.R. “Centralized teleprocessing network design”. *Networks*,1983,vol.13,pp.279-293.

[13] Malik, K. and Yu, G. “A branch and bound algorithm for the capacitated minimum spanning tree problem”. *Networks*,1993, vol.23,pp.525-532.

[14] Papadimitriou, C.H. “The Complexity of the capacitated tree problem”. *Networks*,1978,vol.8,pp.217-230.

[15] Han, J., McMahon, G.B. and Sugden, S.J. “A Node-oriented Branch and Bound Algorithm for the Capacitated Minimum Spanning Tree Problem”. *Proceedings of the 28th IEEE Conference on Local Computer Networks* (LCN)2003,pp.460-469.

[16] Jun Han, Zhaohao Sun, Jinpeng Huai , Xian Li. “An Efficient Node Partitioning Algorithm for the Capacitated Minimum Spanning Tree Problem”. *ACIS-ICIS*,2007,pp.575-580