PERGAMON

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computers & Operations Research 30 (2003) 1931–1944

computers & operations research

www.elsevier.com/locate/dsw

An exact algorithm for the maximum leaf spanning tree problem

Tetsuya Fujie*

Department of Management Science, Kobe University of Commerce, 8-2-1, Gakuen-nishimachi, Nishi-ku, Kobe 651-2197, Japan

Received 1 April 2001; received in revised form 1 April 2002

Abstract

Given a connected graph, the Maximum Leaf Spanning Tree Problem (MLSTP) is to find a spanning tree whose number of leaves (degree-one vertices) is maximum. We propose a branch-and-bound algorithm for MLSTP, in which an upper bound is obtained by solving a minimum spanning tree problem. We report computational results for randomly generated graphs and grid graphs with up to 100 vertices.

Scope and purpose

There exist many applications which can be modeled using graphs. Spanning trees in a graph are often considered since it consists of the minimal set of edges which connect each pair of vertices. The minimum spanning tree problem is a classical and fundamental problem on graphs. In this paper, we consider the maximum leaf spanning tree problem which is to find a spanning tree with the maximum number of leaves (degree-one vertices). This problem has an application in the area of communication networks and circuit layouts. Since the problem is NP-hard, several approximation algorithms have been considered. The purpose of the paper is to propose a branch-and-bound algorithm for the problem. We propose an upper bound which is obtained by solving the minimum spanning tree problem. To the author's knowledge, this is the first exact algorithm to the problem.

Keywords: Branch and bound; Spanning trees; Integer programming

* Tel.: +81-78-794-6161; fax: +81-78-794-6166.
E-mail address: fujie@kobeuc.ac.jp (T. Fujie).

0305-0548/03/$ - see front matter
PII: S0305-0548(02)00117-X

1. Introduction

Consider a spanning tree in a given connected graph G. A vertex is called a *leaf* if it has exactly one incident edge in the spanning tree. In this paper, we consider the Maximum Leaf Spanning Tree Problem (MLSTP), which is to find a spanning tree in G, whose number of leaves is maximum. Several applications of MLSTP can be found in the area of communication networks and circuit layouts [1,2]. For example, let us consider the case of communication networks where the vertices correspond to terminals and the aim is to design a tree-like layout in the network. Then, "leaf terminals" may have lighter work loads than "intermediate terminals" of degree at least two when intermediate terminals have the work on message routing. Hence, in this case, the solution of MLSTP could provide a reasonable layout. A related discussion of this model can be found in [3].

MLSTP is known to be NP-hard [4]. Moreover, it is shown that MLSTP is MAX SNP-hard [5] which implies that there exist $\varepsilon > 0$ and $\alpha > 1$ such that achieving an approximation ratio $(1 + \varepsilon)$ is NP-hard but there is a polynomial time α-approximation algorithm. Lu and Ravi [6,7] developed 3-approximation algorithms and, recently, an improved 2-approximation algorithm was developed by Solis-Oba [8].

MLSTP is equivalent to the Minimum Connected Dominating Set Problem (MCDSP). Here, a subset of vertices is called a connected dominating set if its induced subgraph of the subset is connected and each remaining vertex is adjacent to the subset, and MCDSP is to find a connected dominating set of maximum size. Hence a set of non-leaves of a spanning tree is a connected dominating set and, conversely, a set of vertices outside a connected dominating set is a set of leaves of some spanning tree. Though we know that MCDSP is also NP-hard, the inapproximability result is different. Guha and Khuller [1] showed that the set cover problem is reduced to MCDSP by an approximation preserving reduction. For the set cover problem of n elements in the ground set, Feige [9] showed that achieving an approximation ratio $(1 - \varepsilon)\ln n$ implies NP has $n^{O(\log\log n)}$ deterministic algorithms for any $\varepsilon > 0$.

In this paper, we present a branch-and-bound algorithm for MLSTP. To the author's knowledge, this is the first exact algorithm for MLSTP. We use an integer programming formulation, provided in [10], which will be denoted by (P1). Fernandes and Gouveia [3] gave directed integer programming formulations by replacing each edge by two arcs with opposite directions: One of their formulations is closely related to (P1) (see Section 2). The relaxation problem of (P1) is a minimum spanning tree problem, and hence an upper bound is easily computable. Computational results of the branch-and-bound algorithm will be reported for randomly generated graphs and grid graphs with up to 100 vertices.

The rest of the paper is organized as follows. In Section 2, (P1) is introduced and we make some observations of the spanning tree problem relaxation of (P1). The branch-and-bound algorithm is described in Section 3. In Section 4, we report our computational results for randomly generated graphs and grid graphs. Finally, some concluding remarks are given in Section 5.

2. Formulations and upper bounds

Let $G = (V, E)$ be a connected graph, where V is a set of vertices and E a set of edges. For $i \in V$, let $\delta_G(i)$ denote a set of edges adjacent to i. For a spanning tree $T = (V, E_T)$ in G, the vertex $i \in V$ with $|\delta_T(i)| = 1$ is called a *leaf*. We shall assume $|\delta_G(i)| \geqslant 2$ for $i \in V$: Handling degree-one vertices

($i \in V$ with $|\delta_G(i)| = 1$) will be discussed in Section 3.3. For simplicity, $\delta(i)$ will be used instead of $\delta_G(i)$ if the graph G is clearly understood. For a spanning tree $T = (V, E_T)$ in G, we define its incidence vector $\chi^T = (\chi^T_e \mid e \in E)$ as $\chi^T_e = 1$ if $e \in E_T$; $\chi^T_e = 0$ otherwise. We denote by ST_G a set of all incidence vectors of spanning trees in G.

Let us define a weighted MLSTP. Given a non-negative weight w_i for $i \in V$, the weighed MLSTP is to find a spanning tree T in G, which maximizes $\sum_{i \in L(T)} w_i$ where $L(T)$ is a set of leaves of T. The unweighted MLSTP is associated with $w_i = 1$ for all $i \in V$ and, in the rest of the paper, we will call MLSTP as the unweighted MLSTP. A formulation of the weighted MLSTP is provided as follows:

$$\text{(P1)} \quad \text{maximize} \quad \sum_{i \in V} w_i y_i \tag{1}$$

$$\text{subject to} \quad \boldsymbol{x} \in \mathrm{ST}_G, \tag{2}$$

$$\boldsymbol{x}(\delta(i)) + (|\delta(i)| - 1) y_i \leqslant |\delta(i)| \ (i \in V), \tag{3}$$

$$y_i \in \{0, 1\} \ (i \in V), \tag{4}$$

where $\boldsymbol{x} = (x_e \mid e \in E)$ is a vector of edges and $\boldsymbol{x}(\delta(i)) = \sum_{e \in \delta(i)} x_e$ for $i \in V$. In our formulation (P1), $y_i = 1$ *only if* the vertex i is a leaf of a spanning tree represented by $\boldsymbol{x} \in \mathrm{ST}_G$. Hence the 0–1 vector $\boldsymbol{y} = (y_i \mid i \in V)$ represents a *subset* of leaves of a spanning tree $\boldsymbol{x} \in \mathrm{ST}_G$. Since the weighted MLSTP is a maximization problem and the coefficients w_i ($i \in V$) of the objective function (1) are non-negative, (P1) is a valid formulation of the weighted MLSTP. Formulation (P1) is followed by Fujie [10].

A relaxation problem of (P1) is obtained by relaxing the 0–1 conditions on the variables y_i ($i \in V$). Since (2) and (3) imply $y_i \leqslant 1$ for $i \in V$, we can relax the 0–1 conditions (4) by non-negativity constraints. Then, for any optimal solution of the relaxation problem, the constraint (3) must hold with equality. Hence, the relaxation problem is equivalent to the following problem:

$$\text{($\overline{\text{P1}}$)} \quad \text{maximize} \quad \sum_{i \in V} w_i \frac{|\delta(i)| - \boldsymbol{x}(\delta(i))}{|\delta(i)| - 1}$$

$$= \sum_{i \in V} w_i \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e = \{i,j\} \in E} \left(\frac{w_i}{|\delta(i)| - 1} + \frac{w_j}{|\delta(j)| - 1} \right) x_e$$

$$\text{subject to} \quad \boldsymbol{x} \in \mathrm{ST}_G.$$

($\overline{\text{P1}}$) is a minimum spanning tree problem with weight $w_i/(|\delta(i)| - 1) + w_j/(|\delta(j)| - 1)$ on edge $e = \{i, j\} \in E$, and can be solved efficiently (see e.g. [11]). Note that ($\overline{\text{P1}}$) is equivalent to the Lagrangian relaxation of (P1) with respect to the constraint (3) since this Lagrangian relaxation satisfies the Integral Property [12] (see [10]). We also note that ($\overline{\text{P1}}$) works well as a relaxation problem since we have assumed $|\delta(i)| \geqslant 2$ ($i \in V$).

Here, we make some observations of relaxation ($\overline{\text{P1}}$). Firstly, (P1) remains a valid formulation of the weighted MLSTP even if the constraints (3) are replaced by

$$\boldsymbol{x}(\delta(i)) + (|V| - 2) y_i \leqslant |V| - 1 \quad (i \in V). \tag{5}$$

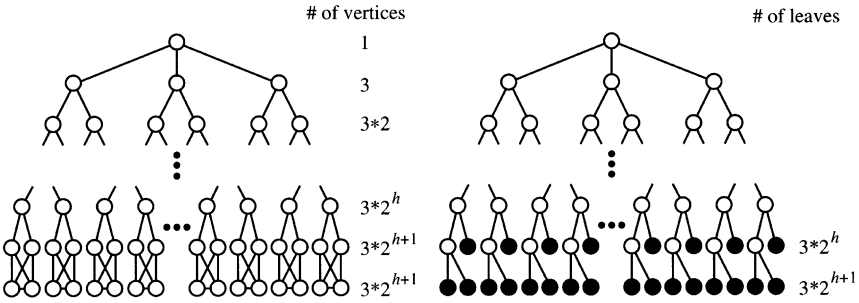

Fig. 1. A 3-regular graph and a spanning tree (black vertex is a leaf).

One of the directed integer programming formulations of Fernandes and Gouveia [3] essentially adopts (5) to relate variables of arcs and vertices. On the other hand, if we use (5) in (P1), the maximum value of $(\overline{\mathrm{P1}})$ for MLSTP is equal to $|V|-1$, which is a trivial upper bound. Secondly, let T_R be an optimal spanning tree of $(\overline{\mathrm{P1}})$. Then we have $\sum_{i\in L(T_R)} w_i \leqslant v(\mathrm{P1}) \leqslant v(\overline{\mathrm{P1}})$, where $v(\cdot)$ denotes the optimal solution value of problem $(\cdot)$. Hence, the relaxation gap $v(\overline{\mathrm{P1}}) - v(\mathrm{P1})$ satisfies

$$v(\overline{\mathrm{P1}}) - v(\mathrm{P1}) \leqslant v(\overline{\mathrm{P1}}) - \sum_{i\in L(T_R)} w_i = \sum_{i\in V'} w_i \frac{|\delta(i)| - |\delta_{T_R}(i)|}{|\delta(i)| - 1},$$

where $V' = \{i \in V \mid 1 < |\delta_{T_R}(i)| < |\delta(i)|\}$. This observation implies that, for example, if any non-leaf $i \notin L$ is full degree (i.e. $|\delta_{T_R}(i)| = |\delta_G(i)|$), T_R is optimal for the weighted MLSTP. Lastly, for MLSTP for an r-regular graph G (i.e. a graph G satisfying $|\delta_G(i)| = r$ for $i \in V$), an explicit upper bound is obtained. Namely, for any spanning tree $\boldsymbol{x} \in \mathrm{ST}_G$, we have

$$\sum_{i\in V} \frac{r - \boldsymbol{x}(\delta(i))}{r-1} = \frac{rn}{r-1} - \frac{\sum_{i\in V} \boldsymbol{x}(\delta(i))}{r-1} = \frac{rn}{r-1} - \frac{2(n-1)}{r-1} = \frac{(r-2)n+2}{r-1}.$$

For 3-regular graphs, the upper bound is $n/2+1$, while Storer [2] gave a lower bound of $n/4+2$. The upper bound is tight: In Fig. 1, we show the tight example. The number of vertices in the graph is

$$\begin{aligned} n &= 1 + 3 + 3(2 + \cdots + 2^h) + 2 \cdot 3 \cdot 2^{h+1} \\ &= 4 + 3 \cdot (2^{h+1} - 2) + 2 \cdot 3 \cdot 2^{h+1} = 9 \cdot 2^{h+1} - 2, \end{aligned}$$

while the number of leaves of the spanning tree is

$$3 \cdot 2^h + 3 \cdot 2^{h+1} = 9 \cdot 2^h = n/2 + 1.$$

For further results on lower bounds, see [13].

3. Algorithm

In this section, we describe a branch-and-bound algorithm based on the formulation (P1).

3.1. Solving subproblems

Recall our formulation (P1) of the weighted MLSTP. In this formulation, the 0–1 vector $\boldsymbol{y}$ represents a subset of leaves of the corresponding spanning tree $\boldsymbol{x} \in \mathrm{ST}_G$. Hence, we can restate the weighted MLSTP as: "Find a *subset* of leaves (leaf subset) of some spanning tree of the maximum weight". This restatement will help us to formulate subproblems. Let us denote a subproblem by $\mathrm{MLSTP}(S_1, S_0, F)$, where (S_1, S_0, F) is a partition of V and any vertex in S_1 must be contained in any leaf subset, none of vertex in S_0 must be contained in any leaf subset, and $F = V \backslash (S_1 \cup S_0)$ is a set of free vertices. The root problem is $\mathrm{MLSTP}(\emptyset, \emptyset, V)$. It can be easily shown that $\mathrm{MLSTP}(S_1, S_0, F)$ has a feasible solution if and only if $G \backslash S_1$ is connected and, for $i \in S_1$, there is an edge that connects i and some vertex in $V \backslash S_1$ [8]. Hence, checking feasibility of $\mathrm{MLSTP}(S_1, S_0, F)$ can be done efficiently.

The subproblem $\mathrm{MLSTP}(S_1, S_0, F)$ is formulated as follows:

$$
\begin{aligned}
(\mathrm{P}(S_1, S_0, F)) \quad \text{maximize} \quad & \sum_{i \in F} w_i y_i + \boldsymbol{w}(S_1) \\
\text{subject to} \quad & \boldsymbol{x} \in \mathrm{ST}_G, \\
& \boldsymbol{x}(\delta(i)) + (|\delta(i)| - 1) y_i \leqslant |\delta(i)| \ \ (i \in F), \\
& \boldsymbol{x}(\delta(i)) \leqslant 1 \ \ (i \in S_1), \\
& y_i \in \{0, 1\} \ \ (i \in F),
\end{aligned}
$$

where $\boldsymbol{w}(S_1) = \sum_{i \in S_1} w_i$. By relaxing the 0–1 constraints, we have a relaxation problem

$$
\begin{aligned}
(\overline{\mathrm{P}(S_1, S_0, F)}) \quad \text{maximize} \quad & \sum_{i \in F} w_i \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e \in E} d_e x_e + \boldsymbol{w}(S_1) \\
\text{subject to} \quad & \boldsymbol{x} \in \mathrm{ST}_G, \\
& \boldsymbol{x}(\delta(i)) \leqslant 1 \ \ (i \in S_1),
\end{aligned}
$$

where

$$
d_e = \begin{cases}
\dfrac{w_i}{|\delta(i)| - 1} + \dfrac{w_j}{|\delta(j)| - 1} & \text{for } e = \{i, j\} \text{ with } i, j \in F, \\
\dfrac{w_i}{|\delta(i)| - 1} & \text{for } e = \{i, j\} \text{ with } i \in F, j \notin F, \\
0 & \text{otherwise.}
\end{cases}
$$

$(\overline{\mathrm{P}(S_1, S_0, F)})$ is solved by finding a minimum spanning tree in $G \backslash S_1$ with costs d_e and then, for $i \in S_1$, connecting i and a vertex $j \in V \backslash S_1$ with the minimum cost $d_{\{i,j\}}$.

Finally, we note that the problem

$$
\begin{aligned}
(\mathrm{P}'(S_1,S_0,F)) \quad \text{maximize} \quad & \sum_{i\in S_1}(W+1)w_i y_i + \sum_{i\in F} w_i y_i - W\boldsymbol{w}(S_1)\\
\text{subject to} \quad & \boldsymbol{x}\in \mathrm{ST}_G,\\
& \boldsymbol{x}(\delta(i)) + (|\delta(i)|-1)y_i \leqslant |\delta(i)| \quad (i\in S_1\cup F),\\
& y_i\in\{0,1\} \ (i\in S_1\cup F)
\end{aligned}
$$

is also a valid formulation of MLSTP(S_1,S_0,F) for large $W>0$ since $y_i=1$ $(i\in S_1)$ holds for any optimal solution. Though its relaxation problem $(\overline{\mathrm{P}'(S_1,S_0,F)})$ is a single minimum spanning tree problem, the algorithm for $(\overline{\mathrm{P}'(S_1,S_0,F)})$ behaves essentially the same as the one for $(\overline{\mathrm{P}(S_1,S_0,F)})$.

3.2. Statement of algorithm

We are now ready to state the branch-and-bound algorithm. For ease of the description, we present a branch-and-bound algorithm for the unweighed MLSTP: Our computational experiments will be done for the unweighted MLSTP and it is easy to extend the branch-and-bound algorithm to the weighted case. For simplicity, the subproblem MLSTP(S_1,S_0,F) is denoted by (S_1,S_0,F).

Algorithm branch_and_bound

1. (**Initialization**) Run the following heuristics:
   BFS: For $v\in V$, apply the breadth first search algorithm rooted at v in the input graph G, to make a spanning tree in G.
   Lu–Ravi: Apply the 3-approximation algorithm of Lu and Ravi [7].
   Solis-Oba: Apply the 2-approximation algorithm of Solis-Oba [8].
   Let LB be the best solution value (the number of leaves) among the three heuristics. If LB$=|V|-1$ then stop. We have had an optimal solution. Otherwise, set $\mathscr{L}:=\{(\emptyset,\emptyset,V)\}$.
2. (**Subproblem selection**) If $\mathscr{L}=\emptyset$, stop. Otherwise, choose $(S_1,S_0,F)\in\mathscr{L}$ in the depth-first fashion and $\mathscr{L}:=\mathscr{L}\backslash\{(S_1,S_0,F)\}$. If $|S_1|+|F|<$ LB then go to 2.
3. (**Checking feasibility**) Check whether there is a spanning tree in G, of which S_1 is a subset of leaves. If not, go to 2.
4. (**Updating a lower bound**) If $|S_1|>$ LB then, for $v\in S_0\cup F$, run the breadth first search algorithm rooted at v in the graph $G\backslash S_1$, to make a spanning tree in G, of which S_1 is a subset of leaves. Let LB be the consequent improved solution value.
5. (**Upper bounding**) Solve $(\overline{\mathrm{P}(S_1,S_0,F)})$ to compute an upper bound UB. If $\lfloor \mathrm{UB}\rfloor \leqslant$ LB, go to 2, where $\lfloor \mathrm{UB}\rfloor$ is the greatest integer not greater than UB.
6. (**Subproblem selection**) Choose $v=\mathrm{argmax}\{|\delta_G(u)|\,|\,u\in F\}$. Let $\mathscr{L}:=\mathscr{L}\cup\{(G,S_1,\bar{S}_0,\bar{F}),(G,\bar{S}_1,S_0,\bar{F})\}$, where $\bar{S}_1=S_1\backslash\{v\}$, $\bar{S}_0=S_0\backslash\{v\}$ and $\bar{F}=F\backslash\{v\}$. Go to 2.

Note that **Lu–Ravi** runs in almost linear time [7] and **Solis-Oba** runs in linear time [8], while the computational time of **BFS** is $O(|V|(|E|+|V|))$. In the approximation algorithms **Lu–Ravi** and **Solis-Oba**, a forest is constructed to obtain the performance ratio, that is, it is proved that any spanning tree containing the forest achieves the ratio. Hence, we construct a spanning tree containing the forest heuristically.

3.3. Handling degree-one vertices

In this subsection, we provide a reformulation of (P1) in the case that an input graph $G=(V,E)$ has some degree-one vertices. To this end, let

$$\widehat{S}_1 = \{i \in V \mid |\delta(i)| = 1\},$$

$$\widehat{S}_0 = \{i \in V \setminus \widehat{S}_1 \mid {}^{\exists} j \in \widehat{S}_1 \text{ s.t. } \{i,j\} \in E\},$$

$$\widehat{F} = V \setminus (\widehat{S}_1 \cup \widehat{S}_0).$$

Since then $y_1 = 1$ $(i \in \widehat{S}_1)$ and $y_i = 0$ $(i \in \widehat{S}_0)$ hold for any feasible solution $\boldsymbol{y}$ of (P1), we can define the problem MLSTP$(\widehat{S}_1, \widehat{S}_0, \widehat{F})$ as a root problem.

4. Computational experiments

Our computational experiments will be done for MLSTP, that is, we will assume $w_i = 1$ for all $i \in V$.

We implemented the branch-and-bound algorithm using PUBB (parallelization utility for branch-and-bound algorithms) developed by Shinano et al. [14] written in C++. Though PUBB is designed for a skeleton of parallel branch-and-bound algorithms, it can also be used for sequential branch-and-bound algorithms. At the user level, we have only to design components, such as lower and upper bounding and branching, to run the sequential and/or the parallel branch-and-bound algorithm. For a detailed description of PUBB, see [14]. In this section, we report results for the sequential branch-and-bound algorithm only. The components of PUBB are written in C and all problems were solved on a Pentium II 300 MHz.

4.1. Random graphs

The purpose of this subsection is to examine the performance of the upper and lower bounds and the limit of the algorithm for randomly generated graphs. Given n and p, we generated a graph of n vertices with density p, where p is the probability that a pair of vertices appears as an edge. We repeated the generation until the graph becomes connected. For each pair of n and p, we generated ten problem instances.

We first report our results of heuristics (see "1. Initialization" in Section 3.2). A part of our results is shown in Table 1. In the table, "Leaves" means the average number of leaves and "Time" the average time in seconds. As the table shows, in our implementation, **BFS** is best for graphs with $p \geqslant 0.3$ and **Lu–Ravi** is best for sparse graphs with $p \leqslant 0.2$. This tendency remains true for other graphs.

Table 2 displays a result of the branch-and-bound algorithm. In the table, 'LB at root' denotes the mean solution value of the heuristics, 'UB at root' the mean upper bound of the input graphs, and 'OPT' the mean optimal solution value. Note that the number of generated subproblems is equal to 1 implies that the optimal solution is obtained at root (without any branching). We omitted results for dense graphs of $p = 0.8$ and 0.9, since most of such dense graphs have stars and optimal solutions are obtained at root. Table 2 shows that neither the upper bound nor the lower bound is very poor.

Table 1
Comparison of heuristics for randomly generated graphs

Graph		**BFS**		**Lu–Ravi**		**Solis-Oba**	
n	p	Leaves	Time	Leaves	Time	Leaves	Time
50	0.1	33.8	0.00	35.4	0.00	34.0	0.00
	0.2	40.4	0.00	40.8	0.00	40.0	0.00
	0.3	44.0	0.00	43.3	0.00	43.2	0.00
	0.4	45.1	0.00	44.4	0.00	44.3	0.00
	0.5	46.4	0.00	45.8	0.00	45.5	0.00
	0.6	47.1	0.01	46.3	0.00	46.1	0.00
	0.7	47.8	0.01	47.0	0.00	47.0	0.00
100	0.1	78.3	0.01	82.0	0.00	77.7	0.00
	0.2	88.4	0.02	88.8	0.00	87.2	0.00
	0.3	92.4	0.03	91.8	0.00	90.9	0.01
	0.4	94.3	0.04	93.2	0.00	93.1	0.01
	0.5	95.8	0.04	94.7	0.00	94.8	0.01
	0.6	96.3	0.05	95.9	0.00	95.5	0.01
	0.7	97.1	0.06	96.3	0.00	96.2	0.01

However, they are unstable so that the number of generated subproblems and the computing time are unstable. Hence, we should improve the upper and lower bounds to solve large problem instances. We also note that the optimal solution values are large and close to the trivial upper bound $n-1$. In fact, Kleitman and West [13] showed that for every connected graph with n vertices and minimum degree at least k has a spanning tree whose number of leaves is at least $(1-b\ln k/k)n$ for large k, where b is a constant exceeding 2.5.

4.2. Grid graphs

In this subsection, we consider grid graphs as an example of graphs with small degrees. The motivation comes from the large optimal solution values for randomly generated graphs.

For positive integers m and n, the grid graph $G_{m\times n}=(V_{m\times n},E_{m\times n})$ is defined as follows:

$$V_{m\times n}=\{(i,j)\mid 1\leqslant i\leqslant m,\ 1\leqslant j\leqslant n\},$$

$$E_{m\times n}=\{\{(i,j),(i,j+1)\}\mid 1\leqslant i\leqslant m,\ 1\leqslant j\leqslant n-1\}$$
$$\cup\{\{(i,j),(i+1,j)\}\mid 1\leqslant i\leqslant m-1,\ 1\leqslant j\leqslant n\}.$$

For $m\geqslant 3$ and $n\geqslant 3$, $G_{m\times n}$ has vertices of degree 4, 3 or 2. The number of degree-4 vertices is $(m-2)(n-2)$, that of degree-3 vertices is $2(m-2)+2(n-2)$, and that of degree-2 vertices is 4. Fig. 2 shows the grid graph $G_{4\times 5}$.

For grid graphs, the upper bound is explicitly calculated for MLSTP.

Lemma 1. *For the grid graph $G_{m\times n}$, an optimal solution value of* $(\overline{\mathrm{P1}})$ *with $w_i=1$ for all $i\in V$ is equal to $2mn/3$. Hence $\lfloor 2mn/3\rfloor$ is a valid upper bound for* MLSTP.

Table 2
Computational results for randomly generated graphs

Graph		LB at root	UB at root	OPT	Generated subproblems			Time (s)		
n	*p*				Min.	Mean	Max.	Min.	Mean	Max.
30	0.1	18.8	21.7	19.7	63	3144.0	22623	0.00	0.25	1.79
	0.2	22.6	25.8	23.9	257	1743.8	6875	0.04	0.25	1.06
	0.3	24.7	27.1	25.5	429	2089.2	5219	0.08	0.39	1.00
	0.4	26.0	27.7	26.6	43	960.8	2747	0.01	0.22	0.63
	0.5	27.0	28.0	27.0	191	346.2	515	0.05	0.10	0.14
	0.6	27.6	28.0	27.9	1	121.4	673	0.00	0.04	0.20
	0.7	28.0	28.0	28.0	1	1.0	1	0.00	0.01	0.01
40	0.1	26.9	31.9	29.0	1435	136451.6	937805	0.24	15.71	107.86
	0.2	32.6	35.9	33.6	2671	20388.6	63771	0.66	5.21	16.34
	0.3	34.0	37.0	35.4	1079	9614.4	20265	0.38	3.15	6.70
	0.4	35.7	37.8	36.5	121	3198.6	7155	0.04	1.24	2.80
	0.5	36.3	38.0	37.0	583	782.8	969	0.30	0.40	0.47
	0.6	37.4	38.0	37.5	1	608.0	1245	0.01	0.37	0.77
	0.7	38.0	38.0	38.0	1	1.0	1	0.01	0.01	0.02
50	0.1	35.7	42.1	38.4	39625	3056245.4	13092423	10.35	480.31	1986.58
	0.2	41.1	45.9	43.4	6545	216824.8	484295	2.60	84.76	188.21
	0.3	44.1	47.0	45.2	861	58893.0	203635	0.42	30.92	101.32
	0.4	45.1	47.8	46.0	9349	16797.4	22123	6.02	11.32	15.62
	0.5	46.5	48.0	47.0	779	1561.4	4167	0.64	1.30	3.28
	0.6	47.1	48.0	47.1	1	1846.8	2159	0.01	1.73	2.12
	0.7	47.8	48.0	48.0	1	57.4	287	0.01	0.07	0.27
60	0.2	50.7	56.0	53.3	25241	685353.6	1375439	14.87	396.83	797.50
	0.3	53.4	57.0	54.9	98553	344504.2	2060863	75.06	259.39	1494.46
	0.4	55.0	57.7	56.0	16169	31173.0	44437	15.23	31.24	46.15
	0.5	56.1	58.0	57.0	1219	2732.0	6149	1.41	3.22	7.08
	0.6	56.9	58.0	57.0	2821	3025.6	3457	3.78	4.07	4.54
	0.7	57.7	58.0	57.9	1	386.2	3167	0.02	0.61	4.93
70	0.3	63.0	67.0	65.0	213273	431577.0	801159	229.53	461.60	849.01
	0.4	64.8	67.9	66.0	39385	55979.4	75377	50.61	74.88	101.51
	0.5	66.0	68.0	66.9	2581	12929.0	75193	4.05	19.79	111.94
	0.6	67.0	68.0	67.1	1	3835.2	4711	0.04	7.12	8.50
	0.7	67.6	68.0	67.9	1	614.4	4313	0.04	1.32	9.21
80	0.4	74.8	77.5	75.9	71815	279422.0	1936327	123.40	461.91	3122.17
	0.5	76.1	78.0	76.7	2915	43423.8	134629	6.02	89.23	276.25
	0.6	76.7	78.0	77.0	5257	5640.0	5997	12.85	13.73	14.50
	0.7	77.4	78.0	77.8	1	1660.2	5927	0.05	4.66	16.87
90	0.5	85.8	88.0	86.4	4851	121648.8	201179	12.62	319.33	524.27
	0.6	86.7	88.0	87.0	7093	7272.8	7553	21.64	22.21	23.03
	0.7	87.0	88.0	87.4	181	4919.6	7709	0.40	16.88	26.50
100	0.5	95.8	98.0	96.3	4809	205794.0	276839	16.25	671.27	918.24
	0.6	96.4	98.0	97.0	8843	9186.0	9541	33.46	35.65	36.97
	0.7	97.1	98.0	97.3	1	6638.6	9411	0.12	29.79	43.52

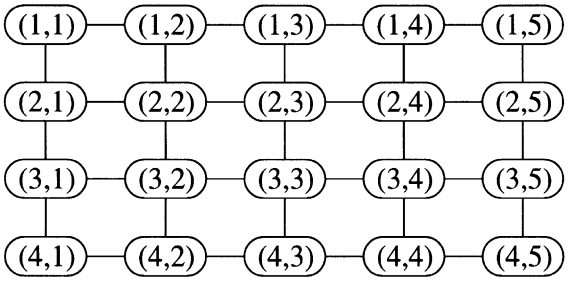

Fig. 2. Grid graph $G_{4\times 5}$.

Proof. Recall that, in the minimum spanning tree problem ($\overline{\mathrm{P1}}$), the coefficient (cost) of an edge (i,j) is equal to $1/(|\delta(i)|-1)+1/(|\delta(j)|-1)$. Hence a minimum spanning tree is constructed by the following steps:

(i) Find a spanning tree in the subgraph induced by the vertices of degree 4. The number of edges in the spanning tree is $(m-2)(n-2)-1$ and each of the edge costs is $\frac{1}{3}+\frac{1}{3}=\frac{2}{3}$.
(ii) For any vertex of degree 3, connect its incident edge to the current spanning tree. The number of vertices is $2(m-2)+2(n-2)$ and each of the new edge costs is $\frac{1}{3}+\frac{1}{2}=\frac{5}{6}$.
(iii) For any vertex of degree 2, connect an edge to the current spanning tree. The number of vertices is 4 and each of the new edge costs is $\frac{1}{2}+\frac{1}{1}=\frac{3}{2}$.

Therefore, the overall cost is

$$\begin{aligned}
&\sum_{i\in V}\frac{|\delta(i)|}{|\delta(i)|-1}-\sum_{e=(i,j)\in E}\left(\frac{1}{|\delta(i)|-1}+\frac{1}{|\delta(j)|-1}\right)x_e\\
&\quad=\left\{(m-2)(n-2)\cdot\frac{4}{3}+(2(m-2)+2(n-2))\cdot\frac{3}{2}+4\cdot 2\right\}\\
&\qquad-\left\{((m-2)(n-2)-1)\cdot\frac{2}{3}+(2(m-2)+2(n-2))\cdot\frac{5}{6}+4\cdot\frac{3}{2}\right\}\\
&\quad=\frac{2mn}{3}
\end{aligned}$$

and completes a proof.

For a lower bound, we have the following lemma.

Lemma 2. *For $m,n\geqslant 4$, $G_{m\times n}$ has a spanning tree whose number of leaves is equal to*

$$mn-\min\left\{2m+(n-4)+\left\lfloor\frac{n-4}{3}\right\rfloor(m-2),2n+(m-4)+\left\lfloor\frac{m-4}{3}\right\rfloor(n-2)\right\}. \tag{6}$$

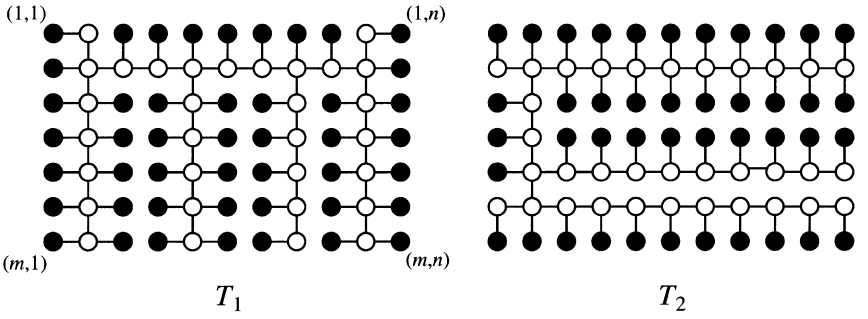

Fig. 3. Constructing T_1 and T_2 for the grid graph $G_{7\times 13}$ (black vertex is a leaf).

Proof. Let T_1 be a spanning tree whose non-leaves are listed below:

$$(1,2),(2,2),\ldots,(m,2),$$

$$(1,n-1),(2,n-1),\ldots,(m,n-1),$$

$$(2,3),(2,4),\ldots,(2,n-2),$$

$$(i,3k+2) \text{ for } i=3,4,\ldots,m,\ k=1,2,\ldots,\left\lfloor\frac{n-4}{3}\right\rfloor.$$

Let T_2 be a spanning tree whose non-leaves are listed below:

$$(2,1),(2,2),\ldots,(2,n),$$

$$(m-1,1),(m-1,2),\ldots,(m-1,n),$$

$$(3,2),(4,2),\ldots,(m-2,2),$$

$$(3\ell+2,j) \text{ for } \ell=1,2,\ldots,\left\lfloor\frac{m-4}{3}\right\rfloor,\ j=3,4,\ldots,n.$$

Fig. 3 displays an example for $G_{7\times 13}$. It is not hard to see that T_1 and T_2 are true spanning trees in $G_{m\times n}$. Moreover, the number of non-leaves in T_1 is equal to

$$2m+(n-4)+\left\lfloor\frac{n-4}{3}\right\rfloor(m-2)$$

and the number of non-leaves in T_2 is equal to

$$2n+(m-4)+\left\lfloor\frac{m-4}{3}\right\rfloor(n-2).$$

Hence the assertion is proved.

Roughly speaking, the upper and lower bounds in Lemmas 1 and 2 are around $2mn/3$. However, neither of them necessarily attains an optimal solution value. In Fig. 4, the left spanning tree is T_1 in the proof of Lemma 2 and the right is an optimal spanning tree for $G_{7\times 7}$. The upper bound is 32 and the lower bound is 27, while the optimal solution value is 29.

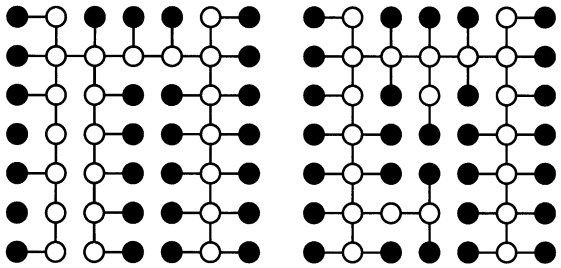

Fig. 4. Grid graph $G_{7\times7}$ and spanning trees (black vertex is a leaf).

Table 3
Computational results for grid graphs

Graph		LB at root				UB at root	OPT	Generated subproblems	Time (s)
m	*n*	**BFS**	**Lu–Ravi**	**Solis-Oba**	Eq. (6)				
3	3	6	6	6	–	6	6	1	0.0
3	4	7	8	8	–	8	8	1	0.0
3	5	9	10	10	–	10	10	1	0.0
3	6	11	12	12	–	12	12	1	0.0
3	7	13	14	14	–	14	14	1	0.0
3	8	15	16	16	–	16	16	1	0.0
3	9	17	18	18	–	18	18	1	0.0
4	4	8	8	8	8	10	9	125	0.0
4	5	10	10	10	11	13	11	311	0.0
4	6	12	14	14	14	16	14	197	0.0
4	7	14	15	16	15	18	16	1473	0.2
4	8	16	18	18	18	21	18	10011	1.2
4	9	18	19	21	21	24	21	5545	0.8
5	5	12	13	13	14	16	14	1545	0.2
5	6	14	17	16	18	20	18	499	0.1
5	7	16	19	18	20	23	20	26383	3.6
5	8	18	21	20	23	26	23	100233	16.6
5	9	20	25	23	27	30	27	34575	6.9
6	6	16	21	19	22	24	22	1327	0.2
6	7	18	24	22	26	28	26	3583	0.7
6	8	20	27	25	30	32	30	10143	2.2
6	9	22	30	28	34	36	34	27061	7.0
7	7	20	27	26	27	32	29	852263	188.5
7	8	22	31	30	33	37	33	4039051	1077.7
7	9	24	35	34	39	42	39	544047	173.0
8	8	24	35	34	38	42	38	61726533	20260.2
8	9	26	40	39	45	48	45	2129061	810.5
9	9	28	45	43	51	54	51	5475435	2597.9

Table 3 displays a computational result for grid graphs, where ‘LB at root’ is the lower bound given in Lemma 2. For $m = 3$, the breadth first search heuristic is applied. As the table shows, the

upper bound is not tight while the lower bound attains the optimum for many graphs. The table also shows that, for the branch-and-bound algorithm, the upper bound is crucial to the computing time.

Finally, we remark that MLSTP is NP-hard even if the graph G is planar with no degree exceeding 4 [4]. On the other hand, to the author's knowledge, it is not known the time complexity of MLSTP for grid graphs, which are still planar with no degree exceeding 4.

5. Concluding remarks

In this paper, we considered MLSTP and its generalization with non-negative weights on vertices. In fact, it can be generalized with weights on edges:

$$\text{maximize} \quad \sum_{i\in V} w_i y_i + \sum_{e\in E} W_e x_e$$

$$\text{subject to} \quad (2), (3), (4),$$

since its relaxation problem is still a minimum spanning tree problem. Note that W_e $(e \in E)$ are not necessarily restricted to be non-negative. Therefore, for the example of the tree-like layout design introduced in Section 1, we can deal with the layout cost.

Acknowledgements

The author is grateful to anonymous referees for many corrections and helpful suggestions.

References

[1] Guha S, Khuller S. Approximation algorithms for connected dominating sets. Proceedings of the Fourth Annual European Symposium on Algorithms, 1996. p. 179–93.
[2] Storer JA. Constructing full spanning trees for cubic graphs. Information Processing Letters 1981;13:8–11.
[3] Fernandes LM, Gouveia L. Minimal spanning trees with a constraint on the number of leaves. European Journal of Operational Research 1998;104:250–61.
[4] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. New York: W. H. Freeman & Co., 1979.
[5] Galbiati G, Maffioli F, Morzenti A. A short note on the approximability of the maximum leaves spanning tree problem. Information Processing Letters 1994;52:45–9.
[6] Lu H, Ravi R. The power of local optimization: approximation algorithms for maximum-leaf spanning tree. Thirtieth Annual Allerton Conference on Communication, Control and Computing, 1992. p. 533–42.
[7] Lu H, Ravi R. Approximating maximum leaf spanning trees in almost linear time. Journal of Algorithms 1998;29: 132–41.
[8] Solis-Oba S. 2-approximation algorithm for finding a spanning tree with maximum number of leaves. Lecture notes in computer science, vol. 1461, 1998. p. 441–52.
[9] Feige U. A threshold of $\ln n$ for approximating set cover. Journal of the ACM 1998;25:634–52.
[10] Fujie T. The maximum-leaf spanning tree problem: formulations and facets. Preprint, 1998. Submitted for publication.
[11] Prim RC. Shortest connection networks and some generalizations. Bell Systems Technical Journal 1957;36: 1389–401.
[12] Geoffrion AM. Lagrangian relaxation for integer programming. Mathematical Programming Study 1974;2:82–114.

[13] Kleitman DJ, West DB. Spanning trees with many leaves. SIAM Journal on Discrete Mathematics 1991;4:99–106.
[14] Shinano Y, Harada K, Hirabayashi R. Control schemes in a generalized utility for parallel branch-and-bound algorithms. In: Proceedings of the 11th International Parallel Processing Symposium, 1997. p. 621–7.

Tetsuya Fujie is a research associate at Department of Management Science, Kobe University of Commerce. He received B.E. and M.E. from Science University of Tokyo, and D.S. from Tokyo Institute of Technology. His research interests include integer programming, combinatorial optimization and global optimization.