Kataoka, Seiji; Yamada, Takeo

Article

Algorithms for the minimum spanning tree problem with resource allocation

Operations Research Perspectives

Provided in Cooperation with:
Elsevier

Suggested Citation: Kataoka, Seiji; Yamada, Takeo (2016) : Algorithms for the minimum spanning tree problem with resource allocation, Operations Research Perspectives, ISSN 2214-7160, Elsevier, Amsterdam, Vol. 3, pp. 5-13, https://doi.org/10.1016/j.orp.2015.12.001

This Version is available at:
https://hdl.handle.net/10419/178266

Operations Research Perspectives 3 (2016) 5–13

Contents lists available at ScienceDirect

Operations Research Perspectives

journal homepage: www.elsevier.com/locate/orp

Algorithms for the minimum spanning tree problem with resource allocation

Seiji Kataoka*, Takeo Yamada

Department of Computer Science, National Defense Academy, Yokosuka, Kanagawa 239-8686, Japan

ARTICLE INFO

Article history:
Received 22 September 2014
Revised 9 December 2015
Accepted 30 December 2015
Available online 13 January 2016

Keywords:
Minimum spanning tree problem
Resource allocation
Trade-off analysis
Branch-and-bound method

ABSTRACT

We formulate the *minimum spanning tree problem with resource allocation* (MSTRA) in two ways, as discrete and continuous optimization problems (d-MSTRA/c-MSTRA), prove these to be $\mathcal{NP}$-hard, and present algorithms to solve these problems to optimality. We reformulate d-MSTRA as the *knapsack constrained minimum spanning tree problem*, and solve this problem using a previously published branch-and-bound algorithm. By applying a 'peg test', the size of d-MSTRA is (significantly) reduced. To solve c-MSTRA, we introduce the concept of *f-fractional solution*, and prove that an optimal solution can be found within this class of solutions. Based on this fact, as well as conditions for 'pruning' subproblems, we develop an enumerative algorithm to solve c-MSTRA to optimality. We implement these algorithms in ANSI C programming language and, through extensive numerical tests, evaluate the performance of the developed codes on various types of instances.

1. Introduction

Numerous applications have been published on the *minimum spanning tree problem* (MST) [1,2] on an undirected graph [3], where each edge is associated with a non-negative *distance*. Here, 'distance' may be cost, time, toll or penalty of each edge in specific applications. In this article we are concerned with a variation of this problem, where each edge may have different 'modes' associated with different pair of distance and 'cost'. Or, we may have edges which can be 'strengthened' by increasing the amount of resources added to these edges. For example, in a city transportation network, we may take a bus or a train to go from one train station to the other, each with respective fare and traveling time. Or, while driving a car we may put more fuel to run faster. We formulate such a combination of the MST and the *resource allocation problem* (RA) [4] as discrete/continuous combinatorial optimization problems, prove these to be NP-hard, develop both of approximate and exact algorithms to solve these problems, and conduct a series of numerical experiments to evaluate the performance of the developed algorithms.

To describe the problem, let $G = (V, E)$ be a connected undirected graph, where V is a finite set of vertices and $E \subseteq V \times V$ is the set of edges. By n and m we denote the numbers of nodes and edges, i.e., $n = |V|$ and $m = |E|$. Associated with each edge $e \in E$ is a cost function $c_e(\cdot)$, which relates r_e, the amount of resource allocated to e, to the cost of this edge. Given a fixed amount R of total resources, our problem is to find a spanning tree of G and a resource allocation on that tree, such that the total cost incurred is minimized over all possible solutions. We formulate this as a discrete, as well as a continuous, optimization problem.

In discrete optimization framework, edges can be either one of 'normal' or 'priority' modes, and the resource requirement and cost of each edge take different values depending on the mode of that edge. If edge e is in normal mode, the amount of resource required is r_e^0, with the corresponding cost c_e^0, while in priority mode these are r_e^1 and c_e^1, respectively. These are related through a binary function $c(\cdot)$ as

$$c_e^0 = c_e(r_e^0), \qquad c_e^1 = c_e(r_e^1) \tag{1}$$

and, we assume

$$c_e^0 \geq c_e^1, \qquad r_e^0 \leq r_e^1, \quad \forall e \in E. \tag{2}$$

That is, in priority mode cost is smaller than in normal mode with an expense of increased resource allocation at that edge.

Let $\mathcal{T}$ denote the set of all the spanning trees in G, and by $T \in \mathcal{T}$ we mean a spanning tree as well as the set of edges constituting T. Resource allocation over T is represented by a binary vector $r = (r_e)_{e \in T}$ with $r_e \in \{r_e^0, r_e^1\}, \forall e \in T$. Then the problem is formulated as the following 'discrete' *minimum spanning tree problem with resource allocation*

* Corresponding author.
E-mail addresses: seiji@nda.ac.jp (S. Kataoka), yamada.144b@gmail.com (T. Yamada).

http://dx.doi.org/10.1016/j.orp.2015.12.001

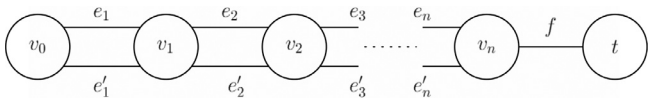


Fig. 1. Graph $G = (V, E)$ for the proof of $\mathcal{NP}$-hardness of c-MSTRA.

d-MSTRA:

$$\text{minimize } z(T, r) := \sum_{e \in T} c_e(r_e) \tag{3}$$

$$\text{subject to } \sum_{e \in T} r_e \le R, \tag{4}$$

$$r_e \in \{r_e^0, r_e^1\}, \quad \forall e \in T, \tag{5}$$

$$T \in \mathcal{T}. \tag{6}$$

Alternatively, the problem may be formulated as a continuous optimization problem. Here, the cost c_e of edge $e \in E$ is a non-increasing function of r_e defined on the continuous interval $[r_e^0, r_e^1]$ as $c_e = c_e(r_e)$. Specifically, we assume this to be linear

$$c_e(r_e) := s_e - \theta_e r_e, \tag{7}$$

and put $c_e^0 := c_e(r_e^0)$ and $c_e^1 := c_e(r_e^1)$ for simplicity. From (2), we have $\theta_e \ge 0$. Thus, the 'continuous' *minimum spanning tree problem with resource allocation* is as follows.

c-MSTRA:

$$\text{minimize} z(T, r) = \sum_{e \in T} c_e(r_e) \tag{8}$$

$$\text{subject to } \sum_{e \in T} r_e \le R, \tag{9}$$

$$r_e^0 \le r_e \le r_e^1, \quad \forall e \in T, \tag{10}$$

$$T \in \mathcal{T}. \tag{11}$$

To prove $\mathcal{NP}$-*hardness* of these problems, we first note that the standard 0–1 knapsack problem

KP:

$$\text{maximize} \quad \sum_{j=1}^{n} p_j x_j$$

$$\text{subject to } \sum_{j=1}^{n} w_j x_j \le R, \quad \forall x_j \in \{0, 1\},$$

is $\mathcal{NP}$-hard.

Theorem 1. *d-MSTRA and c-MSTRA are both $\mathcal{NP}$ -hard.*

Proof ($\mathcal{NP}$-Hardness of d-MSTRA)**.** Corresponding to the knapsack problem let $G = (V, E)$ be with $V = \{v_0, v_1, \ldots, v_n\}$ and $E = \{(v_0, v_1), \ldots, (v_{n-1}, v_n)\}$. For each edge $e = (v_{j-1}, v_j) \in E$ we set $r_e^0 = 0, c_e^0 = p_j$, and $r_e^1 = w_j, c_e^1 = 0$. Then, d-MSTRA is identical to KP, and thus $\mathcal{NP}$-hard.

($\mathcal{NP}$-hardness of c-MSTRA) Given the knapsack problem above, let $G = (V, E)$ be a graph with $V = \{v_0, v_1, \ldots, v_n, t\}$ and $E = \{e_1, e'_1, e_2, e'_2, \ldots, e_n, e'_n, f\}$. Here, e_j and e'_j both connect v_{j-1} and v_j, and $f = (v_n, t)$ (see Fig. 1). The resource and cost are $r^0(e_j) = r^1(e_j) = w_j$ and $c^0(e_j) = c^1(e_j) = M - p_j$ for e_j, $r^0(e'_j) = r^1(e'_j) = 0$ and $c^0(e'_j) = c^1(e'_j) = M$ for e'_j, and $r_f^0 = 0$, $r_f^1 = R$, $c_f^0 = c_f^1 = 0$ for f, where M is a constant satisfying $M > \max_{1 \le j \le n} p_j$. For a solution (T, r) of c-MSTRA on this graph, we introduce a 0–1 variable x_j such that $x_j = 1$ if $e_j \in T$ and $x_j = 0$ otherwise ($j = 1, 2, \ldots, n$). Note that $x_j = 0$ implies $e'_j \in T$. Then, this particular c-MSTRA can be rewritten as

$$\text{minimize} \sum_{j=1}^{n} (M - p_j) x_j$$

$$\text{subject to } \sum_{j=1}^{n} w_j x_j \le R, \quad \forall x_j \in \{0, 1\}.$$

This is equivalent to KP, and thus c-MSTRA is $\mathcal{NP}$-hard. □

The problems formulated above may be regarded as a sort of trade-off analysis of spanning trees with respect to two criteria, the amount of resource consumed $r(T) = \sum_{e \in T} r_e$ and the cost $c(T) = \sum_{e \in T} c_e$ of the tree, provided that r_e and c_e are *a priori* given constants. Hassin and Levin [5] gave a polynomial time approximation scheme, and Yamada et al. [6] gave a branch-and-bound algorithm for such a problem. Trade-off analysis in general is quite standard in scheduling [7–9] and resource allocation [4] problems. If $r(T)$ is regarded as a second objective function, rather than a constraint, we have a multi-objective minimum spanning tree problem [10,11]. An important feature that distinguishes d- and c-MSTRAs from the previous researches is the fact that the coefficient c_e is a function of r_e, thus it can be enhanced by allocating larger amount of resources. To our knowledge, trade-off analysis in this framework is new in this paper.

In Section 2, we discuss d-MSTRA: how this can be reduced to the *knapsack constrained minimum spanning tree problem* (KCMST [6], see also [12]), and how computation can be speeded up by reducing the size of the problem. Sections 3 and 4 explore c-MSTRA and develop solution algorithm to solve this problem to optimality. Finally, in Section 5 a series of numerical experiments are done, both for d- and c-MSTRAs, to examine the behavior of the developed algorithms. Throughout theoretical development in these sections, KCMST plays a key role. Thus, KCMST and its solution algorithm are briefly reviewed in the Appendix for readers' convenience.

2. Solution algorithm for d-MSTRA with problem reduction

In this section we show that d-MSTRA can be reformulated as a KCMST on a 'doubly edged graph'. Furthermore, by applying the 'peg test' the problem is substantially reduced in size. The reduced problem can be solved by SOLVE_KCMST routine [6] much faster than solving the unreduced problem directly.

2.1. d-MSTRA as KCMST

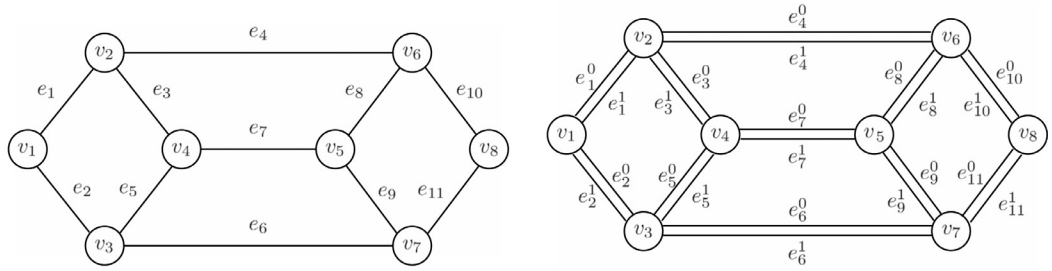
We introduce $\bar{G} = (\bar{V}, \bar{E})$ as the graph which is obtained from G by *doubling* each edge $e \in E$ into edges e^0 and $e^1 \in \bar{E}$, corresponding to normal and priority modes, respectively. Thus, these edges are incident to the identical pair of nodes as $e \in E$, and the resource allocation and cost at e^0 (e^1, resp.) are r_e^0 and c_e^0 (r_e^1 and c_e^1, resp.). We have $\bar{V} = V$ and $|\bar{E}| = 2m$. These graphs are illustrated in Fig. 2 for a planar example. Here we employ the following simplified notation for edges and trees in graph $\bar{G}$. Superscripts in the edges of $\bar{E}$ are usually omitted, unless otherwise needed. Therefore, by $e \in E$ we mean either e^0 or e^1, and c_e may refer to either c_e^0 or c_e^1. $\mathcal{T}$ denotes the set of all the spanning trees in $\bar{G}$. Thus, d-MSTRA can be rewritten as the following

KCMST:

$$\text{minimize } z(T) := \sum_{e \in T} c_e \tag{12}$$

$$\text{subject to } \sum_{e \in T} r_e \le R, \tag{13}$$

$$T \in \mathcal{T}. \tag{14}$$



(a) $G = (V, E)$. (b) $\bar{G} = (\bar{V}, \bar{E})$.

Fig. 2. Graphs G and its duplicate $\bar{G}$.

2.2. Peg test for d-MSTRA

Making use of SOLVE_KCMST [6] to solve KCMST, we can solve d-MSTRA, at least to some extent. However, by reducing the problem size through some preprocessing procedures, we may expect to solve larger instances more quickly. We show here a 'peg test' [13] for this purpose.

We employ the following notations for simplicity's sake: P is equivalently used to denote KCMST, and overbars in $\bar{V}, \bar{E}$, etc. are abbreviated, although we are concerned with $\bar{G} = (\bar{V}, \bar{E})$ throughout this subsection. In the process of SOLVE_KCMST we obtain a (Lagrangian) lower bound $\underline{z}$, a (local search) upper bound $\bar{z}$, and the optimal Lagrangian multiplier $\lambda^\dagger$ (see Appendix and [6]). Let

$$gap := \bar{z} - \underline{z}. \tag{15}$$

For an arbitrary edge $f \in E$, by $\mathrm{P}(f = 1)$ we denote the subproblem of P with a constraint $f \in T$ added, i.e.,
$\mathbf{P}(f = 1)$:

$$\text{minimize} \sum_{e \in T} c_e$$

$$\text{subject to} \sum_{e \in T} r_e \le R,$$

$$f \in T, \qquad T \in \mathcal{T}.$$

Furthermore, $\underline{z}(f = 1)$ denotes a lower bound to this subproblem, and $P(f = 0)$ and $\underline{z}(f = 0)$ are defined analogously with respect to an additional constraint $f \notin T$. Then, the following *peg test theorem* is obvious.

Proposition 1. *Let $T^\star$ denote an optimal solution to P. Then,*

(i) *$\underline{z}(f = 1) > \bar{z}$ implies $f \notin T^\star$, and*
(ii) *$\underline{z}(f = 0) > \bar{z}$ implies $f \in T^\star$.*

In case of (i) or (ii), respectively, f is said to be *fixed-out* or *fixed-in*. Fixed-out edges can never be included in any optimal solution, while those fixed-in must be included in any optimal solution of P. Then, eliminating these fixed edges we obtain a problem of reduced size.

To obtain a lower bound $\underline{z}(f = 1)$ for $f \in E$, we make use of $\lambda^\dagger$ obtained previously, and introduce the relaxation of $\mathrm{P}(f = 1)$ as follows.

$\mathrm{LP}(\lambda^\dagger; f = 1)$:

$$\text{minimize} \sum_{e \in T} c_e + \lambda^\dagger \Big(\sum_{e \in T} r_e - R \Big)$$

$$\text{subject to } f \in T, \qquad T \in \mathcal{T}.$$

Similarly, $\mathrm{LP}(\lambda^\dagger; f = 0)$ is defined, and we obtain a lower bound $\underline{z}(f = 0)$. These are minimum spanning tree problems (with an additional constraint $f \in T$ or $f \notin T$), and can be solved easily as $\mathrm{LP}(\lambda)$. However, to compute $\underline{z}(f = 1)$ and $\underline{z}(f = 0)$ quickly for all $f \in E$, we propose the following approach.

Let $T^\dagger$ be the optimal solution to the Lagrangian relaxation $\mathrm{LP}(\lambda^\dagger)$ (see Appendix). Then, we have $\underline{z} = \sum_{e \in T^\dagger} c_e^\dagger - \lambda^\dagger R$, where for $e \in E$ we define

$$c_e^\dagger := c_e + \lambda^\dagger r_e. \tag{16}$$

Optimal solutions to $\mathrm{LP}(\lambda^\dagger; f = 1)$ and $\mathrm{LP}(\lambda^\dagger; f = 0)$ can be obtained by modifying $T^\dagger$ as follows. For an arbitrary tree edge f in $T^\dagger$, by eliminating f from $T^\dagger$ the set of nodes are partitioned into two disjoint subsets, say V_1 and V_2 ($V = V_1 \cup V_2$, $V_1 \cap V_2 = 08$), which are connected to each of the incident nodes of f through $T^\dagger$, respectively. By $\mathrm{Cut}(T^\dagger, f)$, we denote the set of edges ranging over V_1 and V_2. If we replace f with an edge $e \in \mathrm{Cut}(T^\dagger, f) \setminus \{f\}$, we obtain another tree $T^\dagger - f + e$ with cost $z(T^\dagger - f + e) = \underline{z} + c_e^\dagger - c_f^\dagger$. Then, by *cut optimality property* [3] of $T^\dagger - f + e$, $\mathrm{LP}(\lambda^\dagger; f = 0)$ is solved by finding an edge e, and replacing it with f, such that $c_e^\dagger$ is minimized over $\mathrm{Cut}(T^\dagger, f) \setminus \{f\}$. Thus, with the minimum of this *margin* defined as

$$\Delta(T^\dagger, f) := \min\{c_e^\dagger \mid e \in \mathrm{Cut}(T^\dagger, f), e \ne f\} - c_f^\dagger, \tag{17}$$

we obtain the following optimal objective value to $\mathrm{LP}(\lambda^\dagger; f = 0)$

$$\underline{z}(f = 0) = \underline{z} + \Delta(T^\dagger, f). \tag{18}$$

In case of $f \notin T^\dagger$, adding f to $T^\dagger$ produces an elementary cycle $\mathrm{Cycle}(T^\dagger, f)$ with $\mathrm{Cycle}(T^\dagger, f) \setminus \{f\} \subset T^\dagger$. Then, replacing $f \notin T^\dagger$ with an edge $e \in \mathrm{Cycle}(T^\dagger, f) \setminus \{f\}$, we obtain another tree with cost $z(T^\dagger + f - e) = \underline{z} + c_f^\dagger - c_e^\dagger$, and due to *cycle optimality property* [3] of $T^\dagger + f - e$, $\mathrm{LP}(\lambda^\dagger; f = 1)$ is solved by finding an edge e such that $c_e^\dagger$ is maximized over $\mathrm{Cycle}(T^\dagger, f) \setminus \{f\}$. We define

$$\Theta(T^\dagger, f) := c_f^\dagger - \max\{c_e^\dagger \mid e \in \mathrm{Cycle}(T^\dagger, f), e \ne f\}, \tag{19}$$

and obtain the following optimal value to $\mathrm{LP}(\lambda^\dagger; f = 1)$

$$\underline{z}(f = 1) = \underline{z} + \Theta(T^\dagger, f). \tag{20}$$

To sum up, we have the following *peg test theorem* for problem P.

Theorem 2. *Let $T^\star$ be an optimal solution of P. Then,*

(i) *$f \in T^\dagger$ and $\Delta(T^\dagger, f) > \bar{z} - \underline{z}$ implies $f \in T^\star$ (f is fixed-in), and*
(ii) *$f \notin T^\dagger$ and $\Theta(T^\dagger, f) > \bar{z} - \underline{z}$ implies $f \notin T^\star$ (f is fixed-out).*

3. A Characterization of Optimality for c-MSTRA

We now turn to c-MSTRA. For a spanning tree T of G, $r = (r_e)_{e \in T}$ is said to be a *resource allocation vector* over T. A pair of a spanning tree T and such a vector r over T is a *feasible solution* of c-MSTRA if constraints (9)–(10) are all satisfied. A feasible solution $(T^\star, r^\star)$ is *optimal* if $z(T^\star, r^\star) \le z(T, r)$ for any feasible solution (T, r) of c-MSTRA. We introduce the following.

Definition 1. For edge $f \in E$, a feasible (T, r) is said to be an *f-fractional solution* if $f \in T$ and all edges of T other than f are *extremal*, i.e., $r_e \in \{r_e^0, r_e^1\}, \forall e \in T\ (e \neq f)$.

In f-fractional solution only r_f can take intermediate value in $[r_f^0, r_f^1]$, with all other tree edges either *fixed-low* (i.e., $r_e = r_e^0$), or *fixed-high* ($r_e = r_e^1$). To avoid excessive complication due to ties, we assume that θ_e's are distinct each other for all $e \in E$, i.e.,

$$\theta_e \neq \theta_{e'} \quad \text{for all} e \neq e' \text{in} E.$$

Then, we have the following characterization of optimality of c-MSTRA.

Theorem 3. *There exists an optimal solution $(T^\star, r^\star)$ of c-MSTRA that is f-fractional for some $f \in E$. Furthermore, tree edges other than f are either fixed-low or fixed-high depending on $\theta_e < \theta_f$ or $\theta_e > \theta_f$, respectively, i.e.,*

$$r_e^\star = \begin{cases} r_e^1, & \text{if } \theta_e > \theta_f, \\ r_e^0, & \text{if } \theta_e < \theta_f. \end{cases} \tag{21}$$

Proof. Let $(T^\star, r^\star)$ be an optimal solution of c-MSTRA. We note that for an arbitrary pair of edges e and e' in $T^\star$ with $\theta_e > \theta_{e'}$, either one of these is extremal, i.e., $r_e^\star = r_e^1$ or $r_{e'}^\star = r_{e'}^0$, since otherwise by modifying these to $r_e^\star \to r_e^\star + \epsilon$ and $r_{e'}^\star \to r_{e'}^\star - \epsilon$ for sufficiently small $\epsilon > 0$, the objective value would be improved by $\epsilon(\theta_e - \theta_{e'})$. Thus, if there exist more than one fractional edges, say e and e' in $T^\star$, we must have $\theta_e = \theta_{e'}$. In this case, let $\delta := \min\{r_e^1 - r_e^\star, r_{e'}^\star - r_{e'}^0\}$, and modify the solution to $r_e^\star \to r_e^\star + \delta$ and $r_{e'}^\star \to r_{e'}^\star - \delta$. By doing this, the number of fractional edges is decreased at least by one, and repeating this we will obtain an optimal solution with at most one fractional edge. If no fractional edge remains, we pick up $f := \arg\max_{e \in T^\star}\{\theta_e | e \text{is fixed-low}\}$ as a *degenerate* fractional edge. Thus, we have an optimal f-fractional solution for some $f \in T^\star$.

Finally, if $\theta_e > \theta_f$ and $r_e^\star < r_e^1$, we obtain an improved solution by changing these to $r_e^\star \to r_e^\star + \epsilon$ and $r_f^\star \to r_f^\star - \epsilon$. For optimal $(T^\star, r^\star)$ this is impossible, and thus the former part of (21) is proved. The latter is proved analogously. □

4. An exact algorithm for c-MSTRA

4.1. Initial feasible solution and an upper bound

We note that c-MSTRA is feasible if and only if d-MSTRA is feasible. Indeed, any feasible solution of d-MSTRA is itself feasible to c-MSTRA. On the other hand, if (T, r) is feasible to c-MSTRA, we obtain a feasible solution of d-MSTRA by rounding off all the fractional elements to the corresponding lower bounds, i.e., $r_e \in (r_e^0, r_e^1) \Rightarrow r_e \leftarrow r_e^0$.

Then, our strategy to solve c-MSTRA starts with the optimal solution (T^D, r^D) of d-MSTRA as obtained in Section 2. This solution is feasible to c-MSTRA, but in d-MSTRA the amount of resources actually consumed in optimality is usually smaller than R. Let the residual amount of resource be

$$R_{\text{res}} := R - \sum_{e \in T^D} r_e^D. \tag{22}$$

We note the following.

Proposition 2. *In (T^D, r^D) , if $e \in T^D$ is fixed-low, i.e., $r_e^D = r_e^0$, we have*

$$0 \leq R_{\text{res}} < r_e^1 - r_e^0. \tag{23}$$

Proof. From (4), $0 \leq R_{\text{res}}$ is obvious. If $R_{\text{res}} \geq r_e^1 - r_e^0$, we can increase the value of r_e from $r_e^D = r_e^0$ to r_e^1 and obtain an improved solution to d-MSTRA. This violates the optimality of (T^D, r^D), and thus Proposition 2 is proved. □

This means that the residual R_{res} can be put *in full* to any edge e of T^D if e is fixed-low, without violating the resource limit of e. This gives an improved initial feasible solution to c-MSTRA. Particularly, we pick up $f := \arg\max_{e \in T^D}\{\theta_e | e \text{is fixed-low}\}$, and denote the solution of c-MSTRA obtained this way as (T^C, r^C) with the corresponding upper bound $\bar{z}^C := z(T^C, r^C)$.

4.2. Decomposition

We decompose c-MSTRA into m subproblems and examine these, one by one, for better solutions. For each $f \in E$, let F_f denote the set of all the f-fractional solutions of c-MSTRA, and we introduce subproblem

$\mathbf{P}_f$:

$$\text{minimize} z(T, r) = \sum_{e \in T} c_e(r_e)$$

$$\text{subject to } (T, r) \in F_f,$$

$$\sum_{e \in T} r_e \leq R.$$

By Theorem 3, an optimal solution of c-MSTRA can be obtained as a solution of one of these m subproblems, and to solve P_f we introduce the following *auxiliary graph* G_f.

Definition 2. Graph G_f is obtained from G by *contraction* of edge f, i.e., by eliminating f and identifying two incident nodes of f. Furthermore, edge $e \neq f$ is associated with the following values,

$$(r_e, c_e) = \begin{cases} (r_e^1, c_e^1), & \text{if } \theta_e > \theta_f, \\ (r_e^0, c_e^0), & \text{if } \theta_e < \theta_f. \end{cases} \tag{24}$$

We note that, due to Theorem 3, if (T, r) is an optimal f-fractional solution of P_f with resource r_f allocated to edge f, the non-fractional part $T \setminus \{f\}$ needs to be an optimal solution of KCMST on graph G_f with total amount of resource $R - r_f$. We write this problem $K(G_f, R - r_f)$, and $z_K^\star(G_f, R - r_f)$ denotes the optimal objective value of this problem. Then, since the cost for f is $c_f(r_f)$ and the optimal objective value from G_f is $z_K^\star(G_f, R - r_f)$, the objective value for P_f is

$$z_f^\star(r_f) := c_f(r_f) + z_K^\star(G_f, R - r_f). \tag{25}$$

We also note that $c_f(r_f)$ is a linear, non-increasing function, and $z_K^\star(G_f, R - r_f)$ is a non-decreasing *step function* of $r_f \in [r_f^0, r_f^1]$ as depicted in Fig. 3, and thus P_f is equivalent to finding the optimal allocation of resources between f and G_f, i.e., solving the following.

$\mathbf{P}'_f$:

$$\text{minimize } z_f^\star(r_f)$$

$$\text{subject to } r_f^0 \leq r_f \leq r_f^1.$$

4.3. Exact algorithm

We start with an *incumbent* solution $(T^\natural, r^\natural)$ of c-MSTRA and the corresponding upper bound $\bar{z}^\natural := z(T^\natural, r^\natural)$ of c-MSTRA. This is usually the initial feasible solution obtained earlier, and thus $(T^\natural, r^\natural) := (T^C, r^C)$ and $\bar{z}^\natural := \bar{z}^C$.

Now, we investigate P_f one by one for all $f \in E$. First of all, if for a lower bound $z_K(G_f, R - r_f^0)$ of $K(G_f, R - r_f^0)$

$$c_f(r_f^1) + z_K(G_f, R - r_f^0) \geq \bar{z}^\natural \tag{26}$$

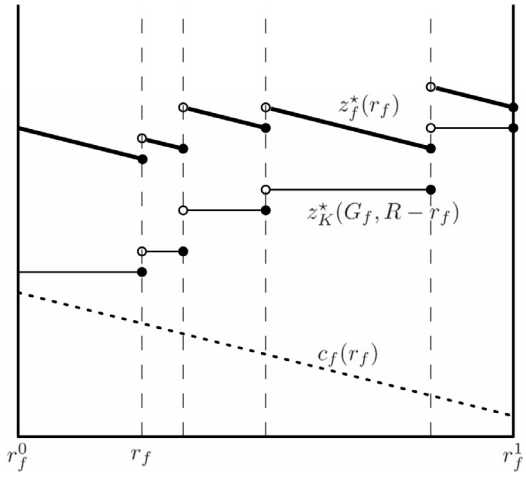


Fig. 3. Function $z_f^\star(r_f)$ (bold) as a sum of $c_f(r_f)$ (dotted) and $z_K^\star(G_f, R-r_f)$ (thin).

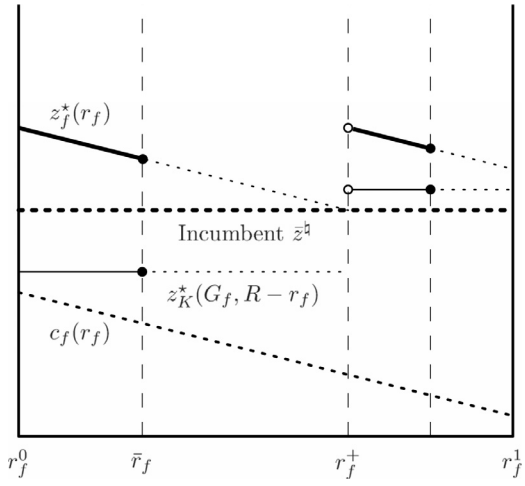


Fig. 4. Larger step-sizes in increasing r.

holds, we can *terminate* P_f, since in this case $z_f^\star(r_f) \geq \bar{z}^\natural$ for all $r_f \in [r_f^0, r_f^1]$, and no better solutions than the incumbent can be expected in this subproblem.

If (26) is not met, we consider a general case of resource allocation $(r_f, R-r_f)$ with $r_f \in [r_f^0, r_f^1]$ allocated to the fractional edge f and the remaining $R-r_f$ to graph G_f. We start with $r_f = r_f^0$, and repeat the following process until $r_f > r_f^1$.

We solve $K(G_f, R-r_f)$ and obtain an optimal solution $T_K^\star(G_f, R-r_f)$ and the optimal $z_K^\star(G_f, R-r_f)$. Correspondingly, we obtain an optimal resource allocation vector $r_K^\star(G_f, R-r_f)$. Note that the actual amount of resources consumed in G_f is the sum of r_e over $T_K^\star(G_f, R-r_f)$, so the residual

$$\bar{r}_f := R - \sum_{e \in T_K^\star(G_f, R-r_f)} r_e \tag{27}$$

can be put on the fractional f.

We see that $r_f \leq \bar{r}_f$, $T_K^\star(G_f, R-r_f') \equiv T_K^\star(G_f, R-\bar{r}_f)$ and $z_K^\star(G_f, R-r_f') \equiv z_K^\star(G_f, R-\bar{r}_f)$ for all $r_f' \in [r_f, \bar{r}_f]$, since throughout this interval $T_K^\star(G_f, R-\bar{r}_f)$ remains optimal. Furthermore, the pair of f and $T_K^\star(G_f, R-\bar{r}_f)$, together with resource allocation $r[f, \bar{r}_f] := (\bar{r}_f, r_K^\star(G_f, R-\bar{r}_f))$, gives an f-fractional solution of c-MSTRA. This is hereafter denoted as $(T^f, r[f, \bar{r}_f])$.

At this stage if

$$c_f(r_f^1) + z_K^\star(G_f, R-\bar{r}_f) \geq \bar{z}^\natural, \tag{28}$$

we can terminate P_f, as in (26). On the other hand, if

$$z_f^\star(\bar{r}_f) < \bar{z}^\natural, \tag{29}$$

$(T^f, r[f, \bar{r}_f])$ is better than the incumbent solution. Thus, we update the incumbent and the corresponding upper bound as $(T^\natural, r^\natural) \leftarrow (T^f, r[f, \bar{r}_f])$ and $\bar{z}^\natural \leftarrow z_f^\star(\bar{r}_f)$.

Finally, we increase r_f beyond $\bar{r}_f$ to move to the next interval of the step function $z_K^\star(G_f, R-r_f)$ by putting $r_f \leftarrow \bar{r}_f + \epsilon$ for sufficiently small $\epsilon > 0$, and repeat the above process all over again. However, in case of $z_f^\star(\bar{r}_f) > \bar{z}^\natural$, r_f can be increased more, because of the following. Let

$$r_f^+ := \bar{r}_f + (z_f^\star(\bar{r}_f) - \bar{z}^\natural)/\theta_f. \tag{30}$$

Then, we have $z_f^\star(r_f') \geq \bar{z}^\natural$ for all $r_f' \in [\bar{r}_f, r_f^+]$, since $z_f^\star(r_f') = c_f(r_f') + z_K^\star(G_f, R-r_f')$ and $z_K^\star(G_f, R-r_f')$ is a non-decreasing function of r_f' (see Fig. 4). Thus, r_f can be increased to $\max\{r_f^+, \bar{r}_f + \epsilon\}$.

To sum up, our algorithm is as follows.

Algorithm **SOLVE_MSTRA**

1. Solve d-MSTRA and obtain an optimal (T^D, r^D) to this discrete optimization problem.
2. Make use of the residual resource to improve (T^D, r^D) and obtain a feasible solution (T^C, r^C) to c-MSTRA. This and $\bar{z}^\natural := z(T^C, r^C)$ serve as an initial incumbent.
3. For each $f \in E$ do:
   - 3-1. If $c_f(r_f^1) + \underline{z}_K(G_f, R-r_f^0) \geq \bar{z}^\natural$, go to next f (Terminate P_f).
   - 3-2. Let $r_f := r_f^0$ and do:
     - 3-2-1. If $r_f > r_f^1$ go to next f (P_f finished).
     - 3-2-2. Solve $K(G_f, R-r_f)$ and obtain an optimal solution $T_K^\star(G_f, R-r_f)$ with the optimal value $z_K^\star(G_f, R-r_f)$ and $\bar{r}_f$ defined by (27).
     - 3-2-3. If $c_f(r_f^1) + z_K^\star(G_f, R-\bar{r}_f) \geq \bar{z}^\natural$, go to next f (Terminate P_f).
     - 3-2-4. If $z_f^\star(\bar{r}_f) < \bar{z}^\natural$, update the incumbent as $(T^\natural, r^\natural) \leftarrow (T^f, r[f, \bar{r}_f])$ and $\bar{z}^\natural \leftarrow z_f^\star(\bar{r}_f)$.
     - 3-2-5. Increment r_f to $r_f \leftarrow \max\{r_f^+, \bar{r}_f + \epsilon\}$ for sufficiently small $\epsilon > 0$, and go to 3-2-1.
4. Output incumbents $(T^\natural, r^\natural)$ and $\bar{z}^\natural$ as optimal to c-MSTRA and stop.

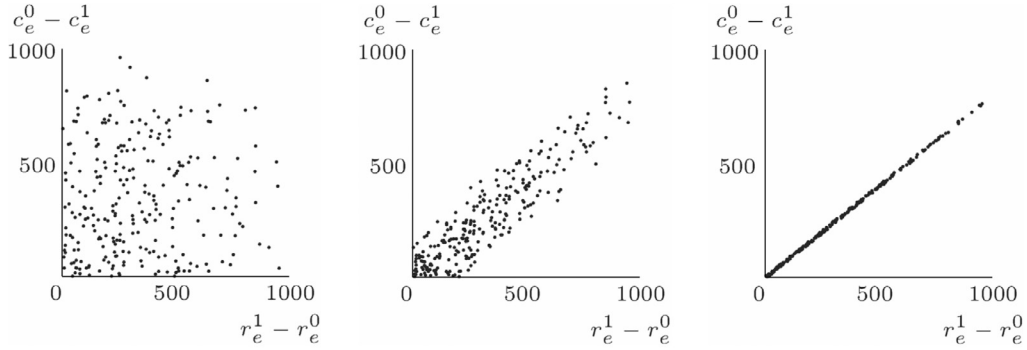
5. Numerical experiments

We have implemented the algorithms to solve both of d- and c-MSTRAs in ANSI C language to optimality, using a DELL Precision T7500 (CPU: Intel Xeon X5680(3.33 GHz) × 2, RAM: 96 GB) with RedHat Enterprise Linux v5.5 and Intel C++ compiler.

5.1. Design of experiments

We prepare planar and complete graphs to represent sparsely and densely edged graphs, respectively. All computations were carried out on instances with edge data generated randomly on these graphs of various sizes. By $P_{n,m}$ we denote a *planar graph* with n nodes and m edges. For example, graph G of Fig. 2 is actually a planar $P_{8,11}$. $K_{n,n(n-1)/2}$ (usually denoted as K_n) is the *complete graph* with n nodes and $n(n-1)/2$ edges.

Resource requirements at each edge are distributed uniformly random over the integer interval $[1, L]$, where L is an experimental parameter which takes either 10^2, 10^3 or 10^4. More precisely, for each $e \in E$ we pick up two integers ξ_e and ξ_e' at random in $[1, L]$,



(a) UNCOR. (b) WEAK. (c) STRONG.

Fig. 5. Correlation types of instances.

Table 1
d-MSTRA experiments: UNCOR ($L = 10^3$, $\alpha = 0.4$).

Graph	$\underline{z}$	$\bar{z}$	%fix0	%fix1	#unfix	$z^\star$	CPU
$P_{100,260}$	20 138.5	20 176.8	76.4	15.0	44.4	20 156.5	0.02
$P_{200,560}$	37 676.5	37 711.1	78.2	14.5	81.6	37 688.8	0.48
$P_{400,1120}$	76 098.8	76 122.8	79.6	15.5	110.1	76 105.6	0.93
$P_{600,1680}$	113 489.5	113 505.2	80.4	16.2	113.0	113 495.3	3.22
$P_{800,2240}$	154 108.5	154 124.1	80.5	16.3	144.6	154 113.8	7.86
$P_{1000,2800}$	190 701.2	190 715.1	80.7	16.4	162.4	190 704.1	12.29
$K_{40,780}$	1134.8	1145.3	96.7	1.7	25.0	1140.6	0.02
$K_{80,3160}$	1282.8	1286.8	98.4	1.0	35.8	1284.8	0.17
$K_{120,7140}$	1339.0	1341.7	99.0	0.7	48.0	1339.9	0.46
$K_{160,12\,720}$	1373.4	1375.1	99.4	0.5	53.8	1373.9	1.12
$K_{200,19\,900}$	1368.9	1369.7	99.4	0.5	41.4	1369.0	4.07

Table 2
d-MSTRA experiments: Correlation compared ($L = 10^3$, $\alpha = 0.4$).

Graph	UNCOR			WEAK			STRONG		
	Gap	%unfix	CPU	Gap	%unfix	CPU	Gap	%unfix	CPU
$P_{100,260}$	38.3	8.54	0.02	26.6	11.33	0.08	234.4	64.69	1.93
$P_{200,560}$	34.6	7.29	0.48	12.8	5.11	0.25	366.6	73.71	61.04
$P_{400,1120}$	24.0	4.92	0.93	18.5	6.71	1.30	961.1	87.54	109.47
$P_{600,1680}$	15.7	3.36	3.22	12.5	5.23	5.25	1505.4	99.99	1734.38
$P_{800,2240}$	15.6	3.23	7.86	10.7	4.26	4.23	2206.8	94.83	1675.32
$P_{1000,2800}$	13.9	2.90	12.29	10.0	3.99	6.50	2763.9	100.00	–
$K_{40,780}$	10.5	1.60	0.02	5.7	0.72	0.01	1.1	0.38	0.01
$K_{80,3160}$	4.0	0.57	0.17	2.9	0.27	0.08	0.5	0.18	2.53
$K_{120,7140}$	2.7	0.34	0.46	2.9	0.14	0.20	0.8	0.25	26.99
$K_{160,12\,720}$	1.7	0.21	1.12	1.5	0.09	0.22	0.8	0.27	191.80
$K_{200,19\,900}$	0.8	0.10	4.07	2.4	0.13	2.25	0.8	0.26	1703.53

–No instances solved within 1800 s.

and put $r_e^0 := \min\{\xi_e, \xi'_e\}$ and $r_e^1 := \max\{\xi_e, \xi'_e\}$. Two more integers c_e^0 and c_e^1 are related to r_e^0 and r_e^1 as follows representing uncorrelated, weakly correlated and strongly correlated cases, respectively, i.e.,

(a) UNCOR: For each $e \in E$ we pick up c_e^0 and c_e^1 at random from $[1, L]$.
(b) WEAK: We put $c_e^k = 0.8r_e^k + \lfloor \eta_e^k/5 \rfloor$ $(k = 0, 1)$.
(c) STRONG: We put $c_e^k = 0.8r_e^k + 0.1L + \lfloor \eta_e^k/100 \rfloor$ $(k = 0, 1)$.

Here η_e^k $(k = 0, 1)$ are distributed uniformly random over $[1, L]$, independent of each other, as well as from other random variables. In all cases, c_e^0 and c_e^1 are rounded to the nearest integers, and if necessary these are swapped such that $c_e^0 \geq c_e^1$ is always satisfied. Fig. 5 plots $(r_e^1 - r_e^0, c_e^0 - c_e^1)$ for $P_{100,260}$.

We note that if tree edges are picked up randomly, expected resource requirement for a spanning tree is $E\{\sum_{e\in T} r_e\} = L(n-1)/2$. Thus, we set the total amount of resource at $R = Ln\alpha$, where α is another parameter that takes either 0.2, 0.4 or 0.6 in our experiments.

5.2. Result of experiments: d-MSTRA

Tables 1 and 2 give the results of experiment for d-MSTRA. Table 1 is the case of UNCOR with $L = 10^3$ and $\alpha = 0.4$, and Table 2 compares the effect of correlation. Here each row represents the average over 10 randomly generated instances, and the columns stand for the following.

$\underline{z}$	:	Lower bound by Lagrangian relaxation.
$\bar{z}$	:	Upper bound improved through local search.
%fix0 (%fix1)	:	Percentage of edges fixed out (in, resp.) by peg test.
#unfix (%unfix)	:	Number (Percentage, resp.) of edges left unfixed.
$z^\star$	:	Optimal objective value.
CPU	:	CPU time in seconds to obtain optimal solution.

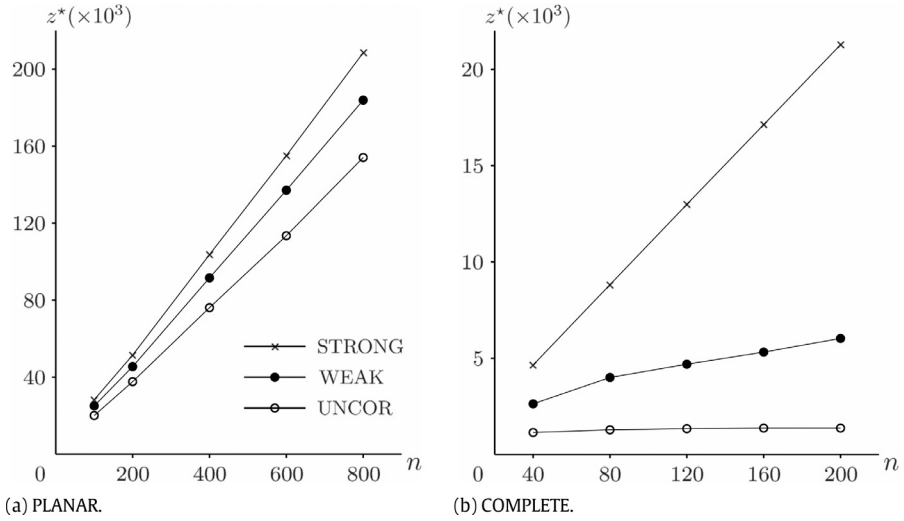


Fig. 6. Objective value $z^\star$ of c-MSTRA as a function of n.

We were able to solve all the instances of Table 1 within a few CPU seconds. From this table we observe the following.

- For the cases tested, we usually have approximate solutions with the objective value close to the lower bounds. For complete graphs optimal solution is often obtained by the heuristic methods, without calling SOLVE_KCMST, especially for large n.

Findings from Table 2 include:

- In UNCOR and WEAK cases, gap $(= \bar{z} - \underline{z})$ decreases with the size of instances. As a consequence, the peg test works effectively to fix many edges either in or out. The ratio (%unfix) of the unfixed edges to the total number of (normal and priority) edges (2 m) is less than 12%, while this is less than 2% in complete graphs. In these cases, the reduced problem is solved within a few seconds.
- However, in planar graphs with STRONG correlation, gap increases rapidly with the problem size, and the peg test is less effective in such a case.

5.3. Result of experiments: c-MSTRA

Tables 3–5 give the result of experiments for each correlation type with $L = 10^3$ and $\alpha = 0.4$, each row again as average over 10 randomly generated instances. The columns stand for the following.

$z^\star$	:	Optimal objective value of c-MSTRA,
$\#P_f$	:	Number of subproblems (P_f's) not terminated either by (26) or (28) at $r_f = r_f^0$,
%term	:	Percentage of subproblems terminated $(100 \cdot (m - \#P_f)/m)$,
#K	:	Total number of KCMSTs solved in solving c-MSTRA,
CPU	:	CPU time in seconds to obtain an optimal solution of c-MSTRA, including the CPU time to solve d-MSTRA.

From these tables we observe the following.

- Irrespective to the size or the correlation type of instance, more than 85% of subproblems are terminated by (26) without solving KCMST. This ratio is often more than 90% in planar graphs, and more than 98% in complete graphs. In complete graphs, the number of remaining P_f's is much smaller than in planar graphs, as well as the number of KCMSTs solved at each instance. This is especially the case in WEAK and STRONG instances.
- Both in planar and complete graphs, the objective value $z^\star$ increases commensurately with n. Correlation between r_e and c_e has little effect on $z^\star$ in planar case, while in the case of

Table 3
c-MSTRA experiments: UNCOR ($L = 10^3$, $\alpha = 0.4$, $R = \alpha Ln$).

Graph	$z^\star$	$\#P_f$	%term	#K	CPU
$P_{100,260}$	20 140.74	25.3	90.3	74.8	2.23
$P_{200,560}$	37 678.31	37.2	93.4	130.7	26.40
$P_{400,1120}$	76 099.86	46.5	95.8	188.7	180.53
$P_{600,1680}$	113 490.15	63.0	96.3	294.9	752.40
$P_{800,2240}$	154 109.80	73.3	96.7	366.6	1929.40
$K_{40,780}$	1136.40	15.5	98.0	29.7	0.39
$K_{80,3160}$	1283.95	21.4	99.3	38.8	5.84
$K_{120,7140}$	1339.21	21.1	99.7	33.8	27.89
$K_{160,12\,720}$	1373.43	24.7	99.8	34.3	99.79
$K_{200,19\,900}$	1368.59	30.0	99.8	35.2	304.63

Table 4
c-MSTRA experiments: WEAK ($L = 10^3$, $\alpha = 0.4$).

Graph	$z^\star$	$\#P_f$	%term	#K	CPU
$P_{100,260}$	25 054.43	27.8	89.3	79.8	1.73
$P_{200,560}$	45 487.58	34.0	93.9	121.0	13.89
$P_{400,1120}$	91 612.93	44.4	96.0	182.8	134.22
$P_{600,1680}$	137 051.04	55.7	96.7	291.2	636.71
$P_{800,2240}$	183 928.86	64.0	97.1	308.6	1039.86
$K_{40,780}$	2628.55	4.5	99.4	7.0	0.12
$K_{80,3160}$	3696.54	2.1	99.9	4.2	2.56
$K_{120,7140}$	4688.30	2.3	99.9	3.8	14.81
$K_{160,12\,720}$	5320.90	1.8	99.9	3.3	49.83
$K_{200,19\,900}$	6033.00	1.4	99.9	3.7	133.22

Table 5
c-MSTRA experiments: STRONG ($L = 10^3$, $\alpha = 0.4$).

Graph	$z^\star$	$\#P_f$	%term	#K	CPU
$P_{100,260}$	28 251.38	26.9	94.2	77.2	1.93
$P_{200,560}$	51 495.78	32.3	96.6	114.3	61.04
$P_{400,1120}$	103 681.39	38.6	97.0	182.9	109.47
$P_{600,1680}$	155 091.17	49.7	97.2	298.7	1734.38
$P_{800,2240}$	208 458.69	61.7	99.8	375.9	1675.32
$K_{40,780}$	4659.00	1.7	99.9	3.1	0.11
$K_{80,3160}$	8821.20	1.2	99.9	1.8	2.53
$K_{120,7140}$	12 991.80	1.2	99.9	1.9	26.99
$K_{160,12\,720}$	17 128.10	1.1	99.9	1.3	191.80
$K_{200,19\,900}$	21 278.00	1.6	99.9	2.3	1703.53

Table 6
c-MSTRA: Sensitivity analysis on L (UNCOR, $\alpha = 0.4$).

Graph	$L = 10^2$		$L = 10^3$		$L = 10^4$	
	$z^\star$	CPU	$z^\star$	CPU	$z^\star$	CPU
$P_{100,260}$	2048.66	0.96	20 140.74	2.23	201 099.23	2.52
$P_{200,560}$	3839.79	6.49	37 678.31	26.31	376 127.38	28.44
$P_{400,1120}$	7752.10	41.28	76 099.86	183.86	759 673.91	266.77
$P_{600,1680}$	11 560.11	177.12	113 490.15	767.69	1 132 909.67	1153.13
$P_{800,2240}$	15 689.80	387.66	154 109.80	1946.32	1 538 421.63	2989.04
$K_{40,780}$	130.97	0.19	1136.40	0.40	11 189.40	0.47
$K_{80,3160}$	164.90	3.74	1283.95	5.86	12 482.59	8.64
$K_{120,7140}$	191.08	22.94	1339.21	29.33	12 842.73	52.11
$K_{160,12\,720}$	215.09	85.49	1373.43	104.86	13 011.32	181.27
$K_{200,19\,900}$	238.45	2743.46	1368.59	292.04	12 783.14	1639.51

Table 7
c-MSTRA: Sensitivity analysis on α (UNCOR, $L = 10^3$, $R = \alpha Ln$).

Graph	$\alpha = 0.2$		$\alpha = 0.4$		$\alpha = 0.6$	
	$z^\star$	CPU	$z^\star$	CPU	$z^\star$	CPU
$P_{100,260}$	39 628.31	1.47	20 140.74	2.26	12 517.86	0.20
$P_{200,560}$	76 023.37	22.21	37 678.31	26.91	23 363.35	1.96
$P_{400,1120}$	153 959.10	183.90	76 099.86	180.88	46 975.47	23.13
$P_{600,1680}$	229 814.34	740.04	113 490.15	756.33	69 640.77	52.04
$P_{800,2240}$	309 794.64	2784.88	154 109.80	1922.89	94 517.33	140.81
$K_{40,780}$	3266.57	0.81	1136.40	0.40	646.43	0.07
$K_{80,3160}$	3727.89	13.58	1283.95	6.31	711.02	2.22
$K_{120,7140}$	4169.93	61.84	1339.21	29.56	722.35	14.05
$K_{160,12\,720}$	4522.27	199.19	1373.43	101.75	735.80	255.68
$K_{200,19\,900}$	4677.59	505.21	1368.59	309.51	747.61	584.91

complete graphs this makes the objective value increase more rapidly with n (see Fig. 6).
- CPU time to solve c-MSTRA increases rapidly with the problem size. Usually the time to solve d-MSTRA to obtain an initial feasible solution of c-MSTRA is negligible, but in STRONG case solving KCMST becomes hard, as observed in [6], which makes both d- and c-MSTRAs even harder to solve.

Table 6 investigates the sensitivity of the range L of random integers on the optimal value $z^\star$ and CPU time in UNCOR type instances with α fixed at 0.4. Objective value increases commensurately with L, and the increase of CPU time is mostly attributed to the increase of the search space of $z^\star$.

Next, Table 7 gives the result of sensitivity analysis on the capacity parameter α, with UNCOR type and $L = 10^3$ fixed. Here the objective value naturally decreases with increasing α. The case of $\alpha = 0.2$ appears to take longer CPU time than the cases of $\alpha \geq 0.4$.

6. Conclusion

In this article, we formulated the minimum spanning tree problem with resource allocation (d- and c-MSTRAs), and gave algorithms to solve these problems to optimality. We proved these to be $\mathcal{NP}$-hard.

We showed that d-MSTRA can be reformulated as a KCMST, which can be solved by the branch-and-bound algorithm we presented previously [6]. Here, we gave a peg test to reduce the problem size of d-MSTRA, often significantly. Using this solution as an initial feasible solution to c-MSTRA, we also gave a solution algorithm for c-MSTRA. This is based on the concept of f-fractional solution. We proved that an optimal solution of c-MSTRA can be found within this class of solutions. Then, the problem was decomposed into a series of subproblems, and we gave an algorithm to examine these subproblems by solving KCMSTs for better f-fractional solutions. In actual computation, many subproblems were eliminated without solving KCMST at all.

We implemented these algorithms in ANSI C language and conducted numerical tests to evaluate the performance of these algorithms on various types of instances. Our algorithm solved instances of considerable size defined on planar as well as on complete graphs exactly. However, to solve larger instances with strong correlation between resource and cost data, more efficient algorithm is required to solve KCMST, since this plays a key role in the algorithm of Section 4.

Appendix. Knapsack constrained minimum spanning tree

Let $G = (V, E)$ be a connected undirected graph with each edge $e \in E$ associated with two numbers, cost c_e and weight w_e, and given $W > 0$ we consider the following *knapsack constrained minimum spanning tree problem*.

KCMST:

$$\text{minimize } z(T) := \sum_{e \in T} c_e \tag{A.1}$$

$$\text{subject to } \sum_{e \in T} w_e \leq W, \tag{A.2}$$

$$T \in \mathcal{T}. \tag{A.3}$$

The *maximization version* of this problem, as discussed in [6], is equivalent to this problem, since for an arbitrary real M and for any spanning tree T of G we have $\sum_{e \in T}(M - c_e) = (n-1)M - \sum_{e \in T} c_e$. In this Appendix, [6] is briefly outlined for readers' convenience.

A.1. Lagrangian relaxation

For a fixed $\lambda \geq 0$, the *Lagrangian relaxation problem* LR(λ):

$$\text{minimize} \sum_{e \in T}(c_e + \lambda w_e) - \lambda W$$

$$\text{subject to } T \in \mathcal{T}$$

is simply the *minimum spanning tree* (MST) problem, which is easily solved [1,2]. Let $\underline{z}(\lambda)$ denote the optimal objective value to this

problem. In addition, $z^\star$ denotes the optimal objective value of KCMST, and T_λ is an optimal solution of LR(λ). Then, the following proposition holds.

Proposition A.1 ([14,13]).

(i) *For an arbitrary* $\lambda \geq 0$, $\underline{z}(\lambda)$ *gives a lower bound to KCMST, i.e.,* $\underline{z}(\lambda) \leq z^\star$.
(ii) $\underline{z}(\lambda)$ *is a piece-wise linear and concave function of* λ *on* $[0, \infty)$.
(iii) *If* $\underline{z}(\lambda)$ *is differentiable at* λ ,

$$\underline{z}'(\lambda) = \sum_{e \in T_\lambda} w_e - W.$$

Thus, $\underline{z}(\lambda)$ is *unimodal* at $\lambda \geq 0$. Then, by the standard *bisection method* the Lagrangian dual
DLR:

$$\text{maximize } \underline{z}(\lambda), \qquad \text{subject to } \lambda \geq 0$$

is solved, and we obtain an optimal $\lambda^\dagger$ and the corresponding 'best' lower bound $\underline{z}_{\text{Lag}} := \underline{z}(\lambda^\dagger)$ to KCMST.

At the same time, each time we compute $\underline{z}(\lambda)$ at $\lambda > \lambda^\dagger$, we have a feasible T_λ and thus an upper bound $z(T_\lambda)$. The minimum of these obtained in the bisection process gives the *Lagrangian upper bound* $\bar{z}_{\text{Lag}}$, and correspondingly the *Lagrangian solution* T_{Lag}.

A.2. Local search

The Lagrangian solution obtained above may further be improved by heuristic algorithms, which is now standard in combinatorial optimization. Here we mention the *2-opt local search* method, which starts with T_{Lag} as an initial solution and repeats to improve the solution as far as possible.

For an arbitrary feasible spanning tree $T \in \mathcal{T}$, its *neighborhood* is defined as follows. For any non-tree edge $f \in E \setminus T$, $T \cup \{f\}$ includes a unique elementary cycle. Let Cycle(T, f) be the set of edges of this cycle. Then, replacing f with an arbitrary edge $e \in$ Cycle$(T, f) \setminus \{f\}$ gives another spanning tree. This is denoted as $T + f - e$, and the *neighborhood* of T is defined as the set of feasible spanning trees obtained from T this way.

Let T_{LS} be the output of the standard 2-opt local search algorithm with respect to the above defined neighborhood. This is referred to as the *local search solution*, and the corresponding local search upper bound is denoted as $\bar{z}_{LS} := z(T_{LS})$.

A.3. SOLVE_KCMST: A branch-and-bound algorithm

We start with the local search solution $T := T_{LS}$, and let this be written explicitly as $T = \{e^1, e^2, \ldots, e^{n-1}\}$. From this we generate a series of *subproblems* in the following way. The ith subproblem is to find an optimal solution that includes $\{e^1, \ldots, e^{i-1}\}$, but does not include $e^i (i = 1, \ldots, n-1)$. Clearly the KCMST is solved if we have solved all these subproblems, and to accomplish this we repeat the same thing recursively with respect to each generated subproblem.

We consider a more general situation where we have a pair of disjoint edge sets I and X. A spanning tree T is said to be (I, X)-*admissible* if it includes all the edges of I, but does not include any of X, i.e., $I \subseteq T$, $T \cap X = 08$. These are referred to as *included* and *excluded* edges, respectively. Then, the subproblem $P(I, X)$ is to find an optimal solution that is feasible and (I, X)-admissible.

We note that the upper and lower bounds introduced previously in Appendices A.1 and A.2 can be easily modified to account for (I, X)-admissibility. Let these be denoted as $\underline{z}(I, X)$ and $\bar{z}(I, X)$ respectively, and $T_{LS}(I, X)$ is the 2-opt solution corresponding to $\bar{z}(I, X)$. In addition, $T^\sharp$ and $z^\sharp$ represent the *incumbent* solution and the corresponding objective value, respectively. Initially these are $T^\sharp := T_{LS}$ and $z^\sharp := \bar{z}_{LS}$.

We construct a recursive routine Solve_$P(I, X)$ to solve subproblem $P(I, X)$ as follows. The input to this routine is the pair (I, X) of disjoint edge sets, and Solve_$P(I, X)$ goes through the following steps.

(1) If $P(I, X)$ is infeasible or $\underline{z}(I, X) \geq z^\sharp$, we *terminate* subproblem $P(I, X)$.
(2) If $P(I, X)$ is solved to optimality, we *update*, if necessary, the incumbent $T^\sharp$ and $z^\sharp$, and terminate the subproblem.
(3) Otherwise, we find an (I, X)-admissible $T_{LS}(I, X)$ by the heuristic method of A.1–A.2. If necessary, we update the incumbent.
(4) We make use of $T_{LS}(I, X)$ to *divide* the problem into a set of mutually disjoint sub-subproblems, and apply Solve_$P(I, X)$ recursively to these sub-subproblems.

In this algorithm, if T_{LS} is written as $I \cup \{e^{k+1}, \ldots, e^{n-1}\}$ with $k := |I|$, the sub-subproblems are generated in (4) as $P(I_i, X_i)$, where $I_i := I \cup \{e^{k+1}, \ldots, e^{i-1}\}$ and $X := X \cup \{e^i\}$ for $i = k + 1, \ldots, n-1$. Then, by calling Solve_$P(08, 08)$, KCMST is solved.

References

[1] Kruskal JB. On the shortest spanning subtree of a graph and the traveling salesman problem. Proc Amer Math Soc 1956;7:8–50.
[2] Prim RC. Shortest connection networks and some generalizations. Bell Syst Tech J 1957;36:1389–401.
[3] Ahuja RK, Magnanti TL, Orlin JB. Network Flows: Theory, Algorithms, and Applications. Englewood Criffs: Prentice-Hall; 1993.
[4] Ibaraki T, Katoh N. Resource Allocation Problems: Algorithmic Approaches. Cambridge, MA: MIT Press; 1988.
[5] Hassin R, Levin A. An efficient polynomial time approximation scheme for the constrained minimum spanning tree problem using matroid intersection. SIAM J Comput 2004;33:261–8.
[6] Yamada T, Watanabe K, Kataoka S. Algorithms to solve the knapsack constrained maximum spanning tree problem. Int J Comput Math 2005;82:23–34.
[7] Değirmenci G, Azizoğlu M. Branch and bound based solution algorithms for the budget constrained discrete time/cost trade-off problem. J Oper Res Soc 2013;64:1474–84.
[8] Martin P.D., Shmoys D.B.. A new approach to computing optimal schedules for the job shop scheduling problem, in: Proceedings 5th International IPCO Conference, Vancouver, Canada, LNCS, Vol. 1084, 1996, 389–403.
[9] Vanhoucke M, Debels D. The discrete time/cost trade-off problem: extensions and heuristic procedures. J Sched 2007;10:311–26.
[10] Ruzika S, Hamacher HW. A Survey on multiple objective minimum spanning tree problems. In: Algorithmics of Large and Complex Networks: Design, Analysis, and Simulation, Bd. 5515; 2009. p. 104–16.
[11] Sourd F, Spanjaard O. A multi-objective branch-and-bound framework. Application to the bi-objective spanning tree problem. INFORMS J Comput 2008;20:472–84.
[12] Hartmann S, Briskorn D. A survey of variants and extensions of the resource–constrained project scheduling problem. European J Oper Res 2010;207:1–14.
[13] Wolsey LA. Integer Programming. New York: John Wiley & Sons; 1998.
[14] Fisher ML. The Lagrangian relaxation method for solving integer programming problems. Manage Sci 2004;50:1861–71.