ELSEVIER

Discrete Applied Mathematics 102 (2000) 245–266

DISCRETE APPLIED MATHEMATICS

Approximation algorithms for some optimum communication spanning tree problems

Bang Ye Wu[a], Kun-Mao Chao[b,*], Chuan Yi Tang[c]

[a] *Chung-Shan Institute of Science and Technology, P.O. Box No. 90008-6-8, Lung-Tan, Taiwan*
[b] *Department of Life Science, National Yang-Ming University, Taipei, Taiwan*
[c] *Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan*

Received 23 January 1998; revised 22 January 1999; accepted 6 July 1999

Abstract

Let $G = (V, E, w)$ be an undirected graph with nonnegative edge length function w and nonnegative vertex weight function r. The optimal product-requirement communication spanning tree (PROCT) problem is to find a spanning tree T minimizing $\sum_{u,v \in V} r(u)r(v)d_T(u,v)$, where $d_T(u,v)$ is the length of the path between u and v on T. The optimal sum-requirement communication spanning tree (SROCT) problem is to find a spanning tree T such that $\sum_{u,v \in V}(r(u) + r(v))d_T(u,v)$ is minimized. Both problems are special cases of the optimum communication spanning tree problem, and are reduced to the minimum routing cost spanning tree (MRCT) problem when all the vertex weights are equal to each other. In this paper, we present an $O(n^5)$-time 1.577-approximation algorithm for the PROCT problem, and an $O(n^3)$ time 2-approximation algorithm for the SROCT problem, where n is the number of vertices. We also show that a 1.577-approximation solution for the MRCT problem can be obtained in $O(n^3)$-time, which improves the time complexity of the previous result.

Keywords: Approximation algorithms; Spanning trees; Network design

1. Introduction

Consider the following network design problem proposed by Hu in [3]. Let $G = (V, E, w)$ be an undirected graph with nonnegative edge length function w. The vertices may represent cities and the edge lengths represent the distances. We are also given the

* Corresponding author.
E-mail addresses: bangye@ms16.hinet.net (B.Y. Wu), kmchao@ym.edu.tw (K. Chao), cytang@cs.nthu.edu.tw (C.Y. Tang)

0166-218X/00/$ - see front matter
PII: S0166-218X(99)00212-7

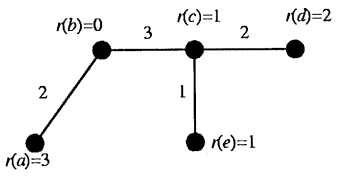
requirements $\lambda(u,v)$ for each pair of vertices, which may represent the number of telephone calls between the two cities. For any spanning tree T of G, the communication cost between two cities is defined to be the requirement multiplied by the path length of the two cities on T, and the communication cost of T is the total communication cost summed over all pairs of vertices. Our goal is to construct a spanning tree with minimum communication cost. That is, we want to find a spanning tree T such that $\sum_{u,v\in V}\lambda(u,v)d_T(u,v)$ is minimized, where $d_T(u,v)$ is the distance between u and v on T.

The above problem is called the *optimum communication spanning tree* (OCT) problem. Let r be a given nonnegative vertex weight function. We consider the following two special cases of the OCT problem in this paper.

- The requirement between a pair of vertices is assumed to be the product of their vertex weights, i.e., $\lambda(u,v)=r(u)\times r(v)$. The *product-requirement communication* (p.r.c.) cost of a tree T is defined by $C_p(T)=\sum_{u,v}r(u)r(v)d_T(u,v)$. Given a graph G, the *optimal product-requirement communication spanning tree* (PROCT) problem is to find a spanning tree T of G such that $C_p(T)$ is minimum among all possible spanning trees. If a vertex v represents a city, then $r(v)$ may be thought of as the population of that city. The communication requirement of a pair of vertices is assumed proportional to the product of the populations of the two cities.
- The requirement between a pair of vertices is defined to be the sum of their vertex weights, i.e., $\lambda(u,v)=r(u)+r(v)$. The *sum-requirement communication* (s.r.c.) cost of a tree T is defined by $C_s(T)=\sum_{u,v}(r(u)+r(v))d_T(u,v)$. Given a graph G, the *optimal sum-requirement communication spanning tree* (SROCT) problem is to find a spanning tree T of G such that $C_s(T)$ is minimum among all possible spanning trees. The SROCT problem may arise in the following situation: For each node in the network, there is an individual message to be sent to every other node and the amount of the message is proportional to the weight of the receiver. With this assumption, the communication cost of a spanning tree T is $\sum_{u,v}r(v)d_T(u,v)$, which is exactly one half of $C_s(T)$.

The two communication costs between a pair of vertices are illustrated in Fig. 1.

When the vertex weights are all equal, e.g., $r(v)=1$ for each vertex v, both the PROCT and the SROCT problems are reduced to the *minimum routing cost spanning tree* (MRCT) problem (also called the *shortest total path length spanning tree* problem). The MRCT problem was shown to be NP-hard in [4] (also listed in [2]). Thus the two problems are also NP-hard. In [6], a 2-approximation algorithm for the



The product-requirement communication cost between vertices a and d is 3*2*(2+3+2)=42.

The sum-requirement communication cost between vertices a and e is (3+1)*(2+3+1)=24.

Fig. 1. The p.r.c. and s.r.c. costs.

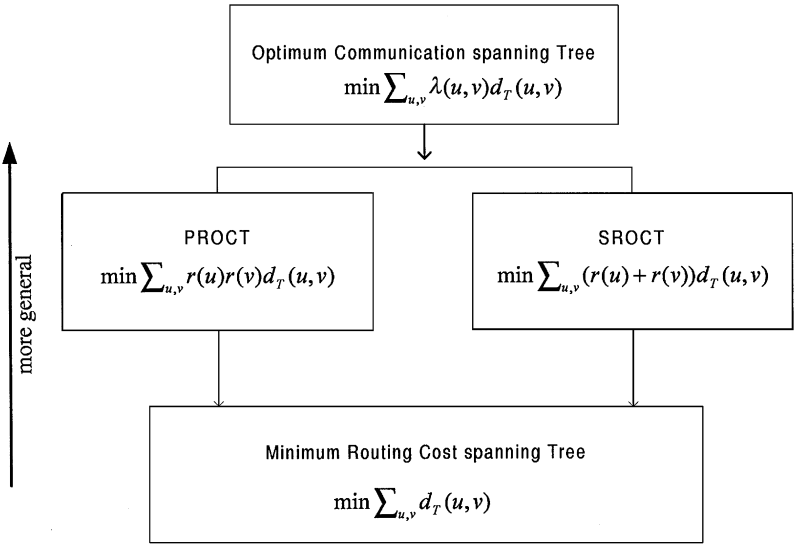


Fig. 2. The relationship of the OCT, PROCT, SROCT, and MRCT problems.

MRCT was presented. Recently, a *polynomial time approximation scheme* (PTAS) for the MRCT problem was proposed and an application to computational biology was discussed in [7]. In this paper, we present a 1.577-approximation algorithm for the PROCT problem, and a 2-approximation algorithm for the SROCT problem. The relationship of the four problems is shown in Fig. 2.

The PTAS for the MRCT problem in [7] was obtained by showing the following properties:

1. The MRCT problem with general inputs is equivalent to the problem with metric inputs (complete graphs in which edge lengths obey the triangle inequality).
2. A k-star is a spanning tree with at most k internal nodes. The minimum routing cost k-star is a $((k+3)/(k+1))$-approximation solution for the metric MRCT problem.
3. For a fixed k, the minimum routing cost k-star on a metric can be found in polynomial time.

In fact, the first and the second properties remain true for the PROCT problem. They can be obtained by straightforward generalizations of the previous results. Consequently, a polynomial time algorithm for the minimum p.r.c. cost k-star is a PTAS for the PROCT problem. However, there is no obvious way to generalize the algorithm for the minimum routing cost k-star to that for the minimum p.r.c. cost k-star. In this paper, we show that the minimum p.r.c. cost 2-star can be found in $O(n^5)$ time by solving a series of min-cut problems. By the result in [7], such a 2-star is a $\frac{5}{3}$-approximation solution of the PROCT problem. With a more precise analysis, we shall show that such a 2-star is in fact a 1.577-approximation solution. This result also improves the approximation ratio of the minimum routing cost 2-star for the MRCT problem from $\frac{5}{3}$ to 1.577.

The previous result in [7] give us an $O(n^4)$ time algorithm for the minimum routing cost 2-star. We shall show that the minimum routing cost 2-star can be solved in $O(n^3 \log n)$ time. Combined with the improvement on the approximation ratio, this leads to an efficient 1.577-approximation algorithm for the MRCT problem. Furthermore, we show that it is possible to find a 1.577-approximation solution for the MRCT problem in $O(n^3)$ time by constructing a special 2-star instead of the minimum routing cost 2-star.

For the SROCT problem, an $O(n^3)$ time 2-approximation algorithm is given in this paper. For any graph, we show that there exists a vertex v such that the *shortest-path tree* rooted at v is a 2-approximation solution of the SROCT problem.

The remaining sections are organized as follows: In Section 2, some definitions and notations are given. The PROCT problem is discussed in Section 3, and the fast approximation algorithm for the MRCT problem is presented in Section 4. The approximation algorithm for the SROCT problem is proposed in Section 5. Finally, we give concluding remarks in Section 6.

2. Preliminaries

In this paper, a graph is a simple, connected and undirected graph. By $G=(V,E,w)$, we denote a graph G with vertex set V, edge set E, and edge length function w. Both the edge length function and the vertex weight function are assumed to be nonnegative. For any graph G, $V(G)$ denotes its vertex set and $E(G)$ denotes its edge set. Let w be an edge length function on a graph G. For a subgraph H of G, we define $w(H)=w(E(H))=\sum_{e\in E(H)} w(e)$. Similarly, let r be a vertex weight function and $U\subset V(G)$. We define $r(U)=\sum_{v\in U} r(v)$ and $r(H)=r(V(H))$ for any subgraph H of G. We shall also use n and R to denote $|V(G)|$ and $r(G)$.

Definition 1. Let $G=(V,E,w)$ be a graph. For $u,v\in V$, $SP_G(u,v)$ denotes a shortest path between u and v on G. The shortest path length is denoted by $d_G(u,v)=w(SP_G(u,v))$.

Definition 2. Let H be a subgraph of G. For a vertex $v\in V(G)$, we use $d_G(v,H)$ to denote the shortest distance from v to H, i.e., $d_G(v,H)=\min_{u\in V(H)} d_G(v,u)$.

The p.r.c. cost and the s.r.c cost of a tree are defined in the previous section. We now define the routing cost of a tree.

Definition 3. For a tree T, the *routing cost* of T is defined by $C(T)=\sum_{u,v\in V(T)} d_T(u,v)$.

Definition 4. Let T be a tree and $e\in E(T)$. Assume X and Y be the two subtrees resulted by deleting e from T. We define the *routing load* on edge e to be $l(T,e)=2|V(X)|\times|V(Y)|$.

Lemma 1. *For a tree T with edge length function w, $C(T)=\sum_{e\in E(T)} l(T,e)w(e)$. In addition, $C(T)$ can be computed in $O(n)$ time, where n is the number of vertices in T.*

Proof.

$$\begin{aligned} C(T) &= \sum_{u,v\in V(T)} d_T(u,v) \\ &= \sum_{u,v\in V(T)} \left(\sum_{e\in SP_T(u,v)} w(e) \right) \\ &= \sum_{e\in E(T)} \left(\sum_{u\in V(T)} |\{v|e\in SP_T(u,v)\}| \right) w(e) \\ &= \sum_{e\in E(T)} l(T,e)w(e). \end{aligned}$$

To compute $C(T)$, we only need to find the routing load on each edge. This can be done in $O(n)$ time by rooting T at any node and traversing T in a postorder sequence. □

Similarly, we define the *product-requirement communication load* (p.r.c. load) and the *sum-requirement communication load* (s.r.c. load) as follows:

Definition 5. Let T be a tree with vertex weight function r and $e\in E(T)$. Assume X and Y be the two subtrees resulted by deleting e from T. The *p.r.c. load* on edge e is defined by $l_p(T,r,e)=2r(X)r(Y)$. The *s.r.c. load* on edge e is defined by $l_s(T,r,e)=2(|V(X)|r(Y)+|V(Y)|r(X))$.

The following corollaries are similar to Lemma 1.

Corollary 2. *For a tree T with edge length function w and vertex weight function r, $C_p(T)=\sum_{e\in E(T)} l_p(T,r,e)w(e)$. In addition, $C_p(T)$ can be computed in $O(n)$ time, where n is the number of vertices in T.*

Corollary 3. *For a tree T with edge length function w and vertex weight function r, $C_s(T)=\sum_{e\in E(T)} l_s(T,r,e)w(e)$. In addition, $C_s(T)$ can be computed in $O(n)$ time, where n is the number of vertices in T.*

For example, let T be the tree in Fig. 1. The p.r.c. load of edge (b,c) is $2(3+0)(1+2+1)=24$ and the p.r.c. cost of T can be computed as follows:

$$\begin{aligned} C_p(T) &= 2\times 3\times 4\times w(a,b)+2\times 3\times 4\times w(b,c) \\ &\quad +2\times 5\times 2\times w(c,d)+2\times 6\times 1\times w(c,e)=172. \end{aligned}$$

The s.r.c. load of edge (b,c) is $2(2)(1+2+1)+2(3+0)(3)=34$ and the s.r.c. cost of T can be computed by

$$C_s(T)=32\times w(a,b)+34\times w(b,c)+26\times w(c,d)+20\times w(c,e)=238.$$

Definition 6. A *metric graph* is a complete graph whose edge lengths satisfy the triangle inequality.

Definition 7. The *metric closure* of a graph G is the complete graph with vertex set $V(G)$ and edge length function δ, where $\delta(u,v)=d_G(u,v)$ for any pair of vertices u and v.

Note that the metric closure of a graph is a metric graph.

Definition 8. Let T be a rooted tree. For any $v\in V(T)$, T_v denotes the subtree with root v.

Definition 9. The *centroid* of a tree T is a vertex $m\in V(T)$ such that if we root T at m, then $|V(T_v)|\leqslant|V(T)|/2$ for any vertex $v\neq m$.

The existence of the centroid of a tree can be easily proved. If we root a tree T at any vertex, then there must exist a vertex m such that $|V(T_m)|>|V(T)|/2$ and $|V(T_v)|\leqslant|V(T)|/2$ for any $v\in V(T_m)\setminus\{m\}$. Since $|V(T)|-|V(T_m)|$ is also no more than $|V(T)|/2$, we conclude that m is the centroid. Similarly, we define the r-centroid of a tree with vertex weight function r.

Definition 10. Let T be a tree with vertex weight function r. The *r-centroid* of a tree T is a vertex $m\in V(T)$ such that if we root T at m, then $r(T_v)\leqslant r(T)/2$ for any vertex $v\neq m$.

For example, both the centroid and the r-centroid of the tree in Fig. 1 are vertex c. If $r(b)=2$ instead of zero, the r-centroid will be vertex b.

3. The PROCT problem

In this section, we discuss the PROCT problem. Let $G=(V,E,w)$ and vertex weight r be the input of the PROCT problem. Our algorithm works as follows:

- Construct the metric closure $\bar{G}$ of G.
- Find the minimum p.r.c. cost 2-star T of $\bar{G}$.
- Transform T into a spanning tree Y of G with $C_p(Y)\leqslant C_p(T)$.

The result for the PROCT problem is stated in the following theorem:

Theorem 4. *There is a* 1.577*-approximation algorithm with time complexity* $O(n^5)$ *for the PROCT problem.*

To prove the correctness of the theorem, we show the following in the next subsections:

- Given any spanning tree T of $\bar{G}$, we can compute from T a spanning tree Y of G such that $C_p(Y) \leqslant C_p(T)$.
- For any $\varepsilon > 0$, if T is a $(1+\varepsilon)$-approximation solution of the ΔPROCT problem with input $\bar{G}$, then Y is a $(1+\varepsilon)$-approximation solution of the PROCT problem with input G, where the ΔPROCT problem has the same definition as the PROCT problem except that the input is always a metric graph.
- The minimum p.r.c. cost 2-star T is a 1.577-approximation solution of the ΔPROCT problem with input $\bar{G}$.
- The overall time complexity is $O(n^5)$.

3.1. A reduction from the general to the metric case

In this subsection, we discuss the transformation algorithm and the related results. The algorithm comes from Wu et al. [7]. It was developed for the MRCT problem, and we show that it also works for the PROCT problem. Let $G = (V, E, w)$ and $\bar{G} = (V, V \times V, \delta)$ be the metric closure of G. Any edge (a, b) in $\bar{G}$ is called a *bad edge* if $(a, b) \notin E$ or $w(a, b) > \delta(a, b)$. Given any spanning tree T of $\bar{G}$, the algorithm first computes the shortest paths for all pairs of vertices. Then a tree Y is constructed by iteratively replacing the bad edges until there exists no bad edge. Since Y has no bad edge, $\delta(e) = w(e)$ for any edge $e \in E(Y)$, and Y can be thought of as a spanning tree of G with the same cost. The algorithm is listed below.

Algorithm Remove_bad
Input: a spanning tree T of $\bar{G}$
Output: a spanning tree Y of G such that $C_p(Y) \leqslant C_p(T)$.
Compute all-pairs shortest paths of G.
(I) **while** there exists a bad edge in T
Pick a bad edge (a, b). Root T at a.
/* assume $SP_G(a, b) = (a, x, \ldots, b)$ and y is the parent of x*/
if b is not an ancestor of x **then**
$Y^* = T \cup (x, b) - (a, b)$; $Y^{**} = Y^* \cup (a, x) - (x, y)$;
else
$Y^* = T \cup (a, x) - (a, b)$; $Y^{**} = Y^* \cup (b, x) - (x, y)$;
endif
if $C_p(Y^*) < C_p(Y^{**})$ **then** $Y = Y^*$ **else** $Y = Y^{**}$ **endif**
(II) $T = Y$
endwhile

The following claim is the same as in [7]. We omit the proof.

Claim 5. *The loop* (I) *is executed at most* $O(n^2)$ *times.*

Claim 6. *Before instruction* (II) *is executed,* $C_p(Y) \leqslant C_p(T)$.

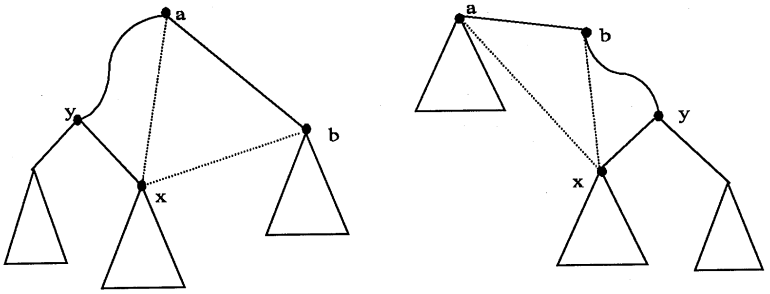


Fig. 3. Remove bad edge (a,b). Case 1 (left) and Case 2 (right).

Proof. For any node v, let $S_v = V(T_v)$. As shown in Fig. 3, there are two cases. Case 2 is identical to Case 1 if we re-root the tree at b and exchange the roles of a and b. Therefore, we only need to show the inequality for Case 1, i.e. $x \in S_a \backslash S_b$.

If $C_p(Y^*) \leqslant C_p(T)$, the result follows. Otherwise, let $U_1 = S_a \backslash S_b$ and $U_2 = S_a \backslash S_b \backslash S_x$. Since in Y^* the distance does not change for any two vertices both in U_1 (or both in S_b), we have

$$C_p(T) < C_p(Y^*) \Rightarrow \sum_{u \in U_1} \sum_{v \in S_b} r(u)r(v)d_T(u,v) < \sum_{u \in U_1} \sum_{v \in S_b} r(u)r(v)d_{Y^*}(u,v).$$

Since for all $u \in U_1$ and $v \in S_b$, $d_T(u,v) = d_T(u,a) + \delta(a,b) + d_T(b,v)$ and $d_{Y^*}(u,v) = d_T(u,x) + \delta(x,b) + d_T(b,v)$, we have

$$\begin{aligned}
&\sum_{u \in U_1} \sum_{v \in S_b} r(u)r(v)(d_T(u,a) + \delta(a,b) + d_T(b,v)) \\
&\quad < \sum_{u \in U_1} \sum_{v \in S_b} r(u)r(v)(d_T(u,x) + \delta(x,b) + d_T(b,v)) \\
&\quad \Rightarrow r(S_b) \sum_{u \in U_1} r(u)d_T(u,a) + r(U_1)r(S_b)\delta(a,b) \\
&\quad < r(S_b) \sum_{u \in U_1} r(u)d_T(u,x) + r(U_1)r(S_b)\delta(x,b) \\
&\quad \Rightarrow \sum_{u \in U_1} r(u)d_T(u,a) + r(U_1)\delta(a,b) < \sum_{u \in U_1} r(u)d_T(u,x) + r(U_1)\delta(x,b).
\end{aligned}$$

Note that $r(S_b) > 0$ since the strict inequality holds. By the definition of the metric closure, we have $\delta(a,b) = \delta(a,x) + \delta(x,b)$, and then

$$\sum_{u \in U_1} r(u)(d_T(u,a) - d_T(u,x)) < -r(U_1)\delta(a,x). \tag{1}$$

Now let us consider the cost of Y^{**}.

$$\begin{aligned}
(C_p(Y^{**}) - C_p(T))/2 = &\sum_{u \in U_2} \sum_{v \in S_x} r(u)r(v)(d_{Y^{**}}(u,v) - d_T(u,v)) \\
&+ \sum_{u \in U_1} \sum_{v \in S_b} r(u)r(v)(d_{Y^{**}}(u,v) - d_T(u,v)).
\end{aligned}$$

Since $d_{Y^{**}}(u,v) \leqslant d_T(u,v)$ for $u \in U_1$ and $v \in S_b$, the second term is not positive. By observing that $d_T(u,v) = d_T(u,x) + d_T(x,v)$ and $d_{Y^{**}}(u,v) = d_T(u,a) + \delta(a,x) + d_T(x,v)$ for any $u \in U_2$ and $v \in S_x$, we have

$$
\begin{aligned}
&(C_{\mathrm{p}}(Y^{**}) - C_{\mathrm{p}}(T))/2 \\
&\quad \leqslant \sum_{u \in U_2} \sum_{v \in S_x} r(u)r(v)(d_T(u,a) + \delta(a,x) - d_T(u,x)) \\
&\quad = r(S_x) \sum_{u \in U_2} r(u)(d_T(u,a) + \delta(a,x) - d_T(u,x)) \\
&\quad = r(S_x) \sum_{u \in U_2} r(u)(d_T(u,a) - d_T(u,x)) + r(U_2)r(S_x)\delta(a,x) \\
&\quad \leqslant r(S_x) \sum_{u \in U_1} r(u)(d_T(u,a) - d_T(u,x)) + r(U_2)r(S_x)\delta(a,x) \qquad (2) \\
&\quad < -r(U_1)r(S_x)\delta(a,x) + r(U_2)r(S_x)\delta(a,x) \qquad (3) \\
&\quad \leqslant 0.
\end{aligned}
$$

Eq. (2) is obtained by observing that $U_1 \backslash U_2 = S_x$ and $d_T(u,a) > d_T(u,x)$ for any $u \in S_x$. Eq. (3) is derived by Eq. (1). Therefore, $C_{\mathrm{p}}(Y^{**}) < C_{\mathrm{p}}(T)$ and the result follows. □

The following lemma comes from the above two claims and the fact that the all-pairs shortest paths can be found in $O(n^3)$ time.

Lemma 7. *Given a spanning tree T of $\bar{G}$, the algorithm Remove_bad constructs a spanning tree Y of G with $C_{\mathrm{p}}(Y) \leqslant C_{\mathrm{p}}(T)$ in $O(n^3)$ time.*

Let $PROCT(G)$ denote the optimum solution of the PROCT problem with input graph G. The above lemma implies that $C_{\mathrm{p}}(PROCT(G)) \leqslant C_{\mathrm{p}}(PROCT(\bar{G}))$. It is easy to see that $C_{\mathrm{p}}(PROCT(G)) \geqslant C_{\mathrm{p}}(PROCT(\bar{G}))$. Therefore, we have the following corollary.

Corollary 8. $C_{\mathrm{p}}(PROCT(G)) = C_{\mathrm{p}}(PROCT(\bar{G}))$.

Corollary 9. *If there is a $(1+\varepsilon)$-approximation algorithm for $\Delta PROCT$ problem with time complexity $O(f(n))$, then there is a $(1+\varepsilon)$-approximation algorithm for $PROCT$ with time complexity $O(f(n) + n^3)$.*

Proof. Let G be the input graph for a PROCT problem. We can construct $\bar{G}$ in time $O(n^3)$ (see e.g. [1]). If there is a $(1+\varepsilon)$-approximation algorithm for the ΔPROCT

problem, we can compute in time $O(f(n))$ a spanning tree T of $\bar{G}$ such that $C_p(T) \leqslant (1+\varepsilon)C_p(PROCT(\bar{G}))$. Using Algorithm Remove_bad, we can then construct a spanning tree Y of G such that $C_p(Y) \leqslant C_p(T) \leqslant (1+\varepsilon)C_p(PROCT(\bar{G})) = (1+\varepsilon)C_p(PROCT(G))$. The overall time complexity is then $O(f(n)+n^3)$. □

3.2. Finding the minimum p.r.c. cost 2-star

In this subsection, we present an algorithm for finding a 2-star T of a metric graph such that $C_p(T)$ is minimum among all possible 2-stars. In the next subsection, we shall show that T is a 1.577 approximation solution for the ΔPROCT problem. Combining the result of Corollary 9, we obtain a 1.577-approximation algorithm for the PROCT problem. We define a notation for 2-stars as follows:

Definition 11. Let $G=(V,E,w)$ be a metric graph. A 2-*star* of G is a spanning tree of G with at most two internal nodes. Assume $x \in X$ and $y \in Y$, and X,Y be a partition of V. We use $2star(x,y,X,Y)$ to denote a 2-star with edge set $\{(x,v) \mid v \in X, v \neq x\} \cup \{(y,v) \mid v \in Y, v \neq y\} \cup \{(x,y)\}$.

The next lemma follows immediately from Lemma 2, and we omit the proof.

Lemma 10. *Let* $T = 2star(x,y,X,Y)$ *and* $R = r(T)$.

$$C_p(T) = 2r(X)r(Y)w(x,y) + 2\sum_{v \in X} r(v)(R-r(v))w(x,v) + 2\sum_{v \in Y} r(v)(R-r(v))w(y,v).$$

Before presenting our algorithm, we briefly explain why the minimum p.r.c. cost 2-star (or even k-star) cannot be found by the algorithm in [7]. Any k-star can be described by a triple $(S,\tau,\mathscr{L})$, where $S=\{v_1,\ldots,v_k\} \subseteq V$ is the set of k distinguished vertices which may have degree more than one, τ is a spanning tree topology on S, and $\mathscr{L}=(L_1,\ldots,L_k)$, where $L_i \subseteq V \backslash S$ is the set of vertices connected to vertex $v_i \in S$. Let $A=(n_1,\ldots,n_k)$ be a nonnegative k-vector (a vector whose components are k nonnegative integers) such that $\sum_{i=1}^{k} n_i = n-k$. We say that a k-star $(S,\tau,\mathscr{L})$ has the configuration (S,τ,A) if $n_i = |L_i|$ for all $1 \leqslant i \leqslant k$. For a fixed k, the total number of configurations is $O(n^{2k-1})$ since there are $\binom{n}{k}$ choices for S, k^{k-2} possible tree topologies on k vertices, and $\binom{n-1}{k-1}$ possible such k-vectors. Note that any two k-stars with the same configuration have the same routing load on their corresponding edges.

Any vertex v in $V \backslash S$ that is connected to a node $s \in S$ contributes to the (standard) routing cost a term of $w(v,s)$ multiplied by its routing load of $2(n-1)$. Since all these routing loads are the same, the best way of connecting the vertices in $V \backslash S$ to nodes

in S, is obtained by finding a minimum-cost way of matching up the nodes of $V\backslash S$ to those in S which obeys the degree constraints on the nodes of S imposed by the configuration, and the costs are the edge weights w. This problem can be solved in polynomial time for a given configuration (by a straightforward reduction to an instance of minimum-cost perfect matching).

The reason why we cannot find the minimum p.r.c. cost k-star by the above method is that we do not know how to find (in polynomial time) the best way to connect the vertices in $V \setminus S$ to nodes in S even for a fixed configuration. Two k-stars with the same configuration may have *different p.r.c. loads* on their corresponding edges. If we modify the definition of the configuration so that $n_i = r(L_i)$, then it may be possible to find the best leaf connection for a fixed configuration in polynomial time, but the number of configurations will be exponential.

Another question is whether the minimum p.r.c. cost k-star can be found by an incremental method similar to the one in [7]. Let us focus on the case $k=2$. For fixed x and y, let X_i and Y_i be the vertex sets such that $2star(x, y, X_i, Y_i)$ is the minimum routing cost 2-star with exact i leaves connected to x for $i = 0, 1, \ldots, n-2$. The key point of the incremental method in [7] is the following property: There always exists a vertex $v \in Y_i$ such that $X_{i+1} = X_i \cup \{v\}$. Therefore, instead of solving many assignment problems, all X_i can be found one by one. However, the property does not hold for the p.r.c. cost 2-star. For example, assume $X_1 = \{v_1\}$ and $Y_1 = \{v_2, v_3\}$. All vertex weights on x, y, v_1 are small, and $r(v_2) = r(v_3) = a$ is a large number. The vertex weights are set in such a way that the p.r.c. load on edge (x, y) will be very large if $\{v_1, v_2\}$ or $\{v_1, v_3\}$ is the set of leaves connected to x. The large load will force $X_2 = \{v_2, v_3\}$, and this is a counterexample of the above property.

Now let us turn to our algorithm for the minimum p.r.c. cost 2-star. If for any specified x and y we can find the best partition X and Y in $O(f(n))$ time, then we can solve the minimum p.r.c. cost 2-star problem in $O(n^2 f(n))$ time by trying all possible vertex pairs for x and y. To find the best partition for a specified pair of vertices x and y, we construct an auxiliary graph $H_{x,y}$, which is an undirected complete graph with vertex set V and edge length function h. The edge length h is defined as follows:

1. $h(x, y) = 2r(x)r(y)w(x, y)$.
2. $h(x, v) = 2r(v)(R - r(v))w(y, v) + 2r(v)r(x)w(x, y)$, and
   $h(y, v) = 2r(v)(R - r(v))w(x, v) + 2r(v)r(y)w(x, y)$ for any vertex $v \notin \{x, y\}$.
3. $h(u, v) = 2r(u)r(v)w(x, y)$ for all $u, v \notin \{x, y\}$.

Let V_1 and V_2 be two subsets of V. We say that (V_1, V_2) is an x–y cut of $H_{x,y}$ if (V_1, V_2) forms a partition of V and $x \in V_1$ and $y \in V_2$. The cost of an x–y cut (V_1, V_2) is defined to be $h(V_1, V_2) = \sum_{u \in V_1, v \in V_2} h(u, v)$. The following lemma comes directly from the above construction. Note that the 2-star is defined on the metric graph G and the cost of the cut is defined on the auxiliary graph $H_{x,y}$.

Lemma 11. *If (V_1, V_2) is an x–y cut of graph $H_{x,y}$, then $h(V_1, V_2) = C_p(2star(x, y, V_1, V_2))$.*

Proof.

$$
\begin{aligned}
h(V_1, V_2) &= \sum_{u \in V_1, v \in V_2} h(u, v) \\
&= \sum_{v \in V_2 - \{y\}} h(x, v) + \sum_{u \in V_1 - \{x\}} h(u, y) + \sum_{u \in V_1 - \{x\}} \sum_{v \in V_2 - \{y\}} h(u, v) + h(x, y) \\
&= \sum_{v \in V_2 - \{y\}} (2r(v)(R - r(v))w(y, v) + 2r(v)r(x)w(x, y)) \\
&\quad + \sum_{u \in V_1 - \{x\}} (2r(u)(R - r(u))w(x, u) + 2r(u)r(y)w(x, y)) \\
&\quad + \sum_{u \in V_1 - \{x\}} \sum_{v \in V_2 - \{y\}} 2r(u)r(v)w(x, y) + 2r(x)r(y)w(x, y) \\
&= \sum_{v \in V_2 - \{y\}} 2r(v)(R - r(v))w(y, v) + \sum_{v \in V_1 - \{x\}} 2r(v)(R - r(v))w(x, v) \\
&\quad + 2r(V_1)r(V_2)w(x, y) \\
&= C_{\mathrm{p}}(2star(x, y, V_1, V_2)). \qquad \square
\end{aligned}
$$

The above lemma implies that the minimum p.r.c. cost 2-star can be found by solving the minimum cut problems on $O(n^2)$ different auxiliary graphs. Since the minimum cut of a graph can be found in $O(n^3)$ (e.g. [1]), we have the following lemma:

Lemma 12. *The minimum p.r.c. cost* 2*-star can be found in* $O(n^5)$ *time.*

3.3. The approximation ratio

In this subsection, we shall investigate the approximation ratio of the minimum p.r.c. cost 2-star for the ΔPROCT problem. Let $G=(V,E,w)$ and r be the input metric graph and the vertex weight of a ΔPROCT problem, respectively. Also let T be the optimal spanning tree of the ΔPROCT problem and m be the r-centroid of T. Root T at its r-centroid m and let $\frac{1}{3} < q < 0.5$ be a real number to be determined later. Consider all possible vertices x such that $r(T_x) \geqslant qR$ and $r(T_u) < qR$ for any $u \in V(T_x)\backslash\{x\}$. By the definition of the r-centroid, there are three cases:

- there are two such vertices a and b;
- there is only one such vertex $a \neq m$;
- m is the only one such vertex.

For each case, we select two vertices. For the first case, a and b are selected. For the second case, a and m are selected, and the third case can be thought of as a special case in which the two vertices are both m. Without loss of generality, assume the two vertices be a and b, and $M = SP_T(a,b) = (a = m_1, m_2, \ldots, m_k = b)$ be the path on T.

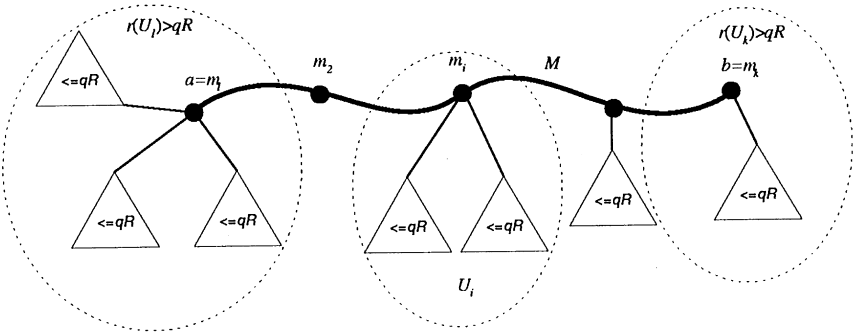


Fig. 4. Notations for showing the approximation ratio.

Also let U_i be the set of vertices which are connected to M at m_i for $i = 1, 2, \ldots, k$. The notations are illustrated in Fig. 4. We have the following lemma:

Lemma 13. $C_p(T) \geqslant 2(1-q)R\sum_x r(x)d_T(x,M) + 2q(1-q)R^2 w(M)$.

Proof. For any vertex x, let $SB(x) = \{u | SP_T(x,u) \cap M = \emptyset\}$. Note that $r(SB(x)) < qR$ by the construction of M. If $x \in U_i$ and $y \in U_j$, define $g(x,y) = d_T(m_i, m_j)$. Then,

$$\begin{aligned}
C_p(T) &= \sum_x \sum_y r(x)r(y)d_T(x,y) \\
&\geqslant \sum_x \sum_{y \notin SB(x)} r(x)r(y)d_T(x,y) \\
&= \sum_x \sum_{y \notin SB(x)} r(x)r(y)\{d_T(x,M) + d_T(y,M) + g(x,y)\} \\
&= 2\sum_x \sum_{y \notin SB(x)} r(x)r(y)d_T(x,M) + \sum_x \sum_{y \notin SB(x)} r(x)r(y)g(x,y) \\
&\geqslant 2(1-q)R\sum_x r(x)d_T(x,M) + \sum_x \sum_{y \notin SB(x)} r(x)r(y)g(x,y).
\end{aligned}$$

Without loss of generality, we assume $r(U_1) \geqslant r(U_k)$. For the second term,

$$\begin{aligned}
&\sum_x \sum_{y \notin SB(x)} r(x)r(y)g(x,y) \\
&= 2\sum_{i<j} r(U_i)r(U_j)d_T(m_i,m_j) \\
&\geqslant 2r(U_1)r(U_k)d_T(m_1,m_k) + 2\sum_{i=2}^{k-1} r(U_i)(r(U_1)d_T(m_1,m_i) + r(U_k)d_T(m_k,m_i)) \\
&\geqslant 2r(U_1)r(U_k)w(M) + 2\sum_{i=2}^{k-1} r(U_i)r(U_k)w(M) \\
&= 2w(M)r(U_k)(R - r(U_k)).
\end{aligned}$$

By the construction of M and $q < 0.5$, we have $r(U_1) \geqslant r(U_k) \geqslant qR$, and then $qR \leqslant r(U_k) \leqslant (1-q)R$. Thus $r(U_k)(R - r(U_k)) \geqslant q(1-q)R^2$ and this completes the proof. □

Construct two 2-stars $T^* = 2star(a, b, V - U_k, U_k)$ and $T^{**} = 2star(a, b, U_1, V - U_1)$. We claim that one of the two 2-stars is an approximation solution with approximation ratio $\max\{1/(1-q), (1-2q^2)/(2q(1-q))\}$. First, we show the following lemma:

Lemma 14. $C_p(T^*) + C_p(T^{**}) \leqslant 4R \sum_{v \in V} r(v) d_T(v, M) + 2(1-2q^2)R^2 w(M)$.

Proof. By Lemma 10,

$$\begin{aligned} C_p(T^*) &= 2 \sum_{v \notin U_k} r(v)(R - r(v))w(v, a) + 2 \sum_{v \in U_k} r(v)(R - r(v))w(v, b) \\ &\quad + 2r(U_k)(R - r(U_k))w(a, b) \\ &\leqslant 2R \sum_{v \notin U_k} r(v)w(v, a) + 2R \sum_{v \in U_k} r(v)w(v, b) \\ &\quad + 2r(U_k)(R - r(U_k))w(a, b). \end{aligned}$$

Similarly,

$$C_p(T^{**}) \leqslant 2R \sum_{v \in U_1} r(v)w(v, a) + 2R \sum_{v \notin U_1} r(v)w(v, b) + 2r(U_1)(R - r(U_1))w(a, b).$$

By the triangle inequality, $w(x, y) \leqslant d_T(x, y)$ for any vertices x and y. Therefore, for any vertex $v \in U_1$, $w(v, a) \leqslant d_T(v, M)$. Similarly, $w(v, b) \leqslant d_T(v, M)$ for any vertex $v \in U_k$. For any vertex $v \notin U_1 \cup U_k$, by the triangle inequality, $w(v, a) + w(v, b) \leqslant 2d_T(v, M) + w(M)$. We have

$$\begin{aligned} C_p(T^*) + C_p(T^{**}) &\leqslant 4R \sum_{v \in V} r(v) d_T(v, M) + 2R \sum_{v \notin U_1 \cup U_k} r(v)w(M) \\ &\quad + 2r(U_k)(R - r(U_k))w(a, b) + 2r(U_1)(R - r(U_1))w(a, b) \\ &= 4R \sum_{v \in V} r(v) d_T(v, M) + 2(R^2 - r(U_1)^2 - r(U_k)^2)w(M) \\ &\leqslant 4R \sum_{v \in V} r(v) d_T(v, M) + 2(1-2q^2)R^2 w(M). \end{aligned}$$ □

Lemma 15. *There is a 2-star which is a 1.577-approximation solution of the $\Delta PROCT$ problem.*

Proof. Trivially, T^* and T^{**} are both 2-stars. By Lemma 14, we have

$$\min\{C_p(T^*), C_p(T^{**})\} \leqslant 2R \sum_{v \in V} r(v) d_T(v, M) + (1-2q^2)R^2 w(M).$$

By Lemma 13, the approximation ratio is $\max\{1/(1-q), (1-2q^2)/(2q(1-q))\}$ in which $\frac{1}{3} < q < \frac{1}{2}$. By setting $q = (\sqrt{3} - 1)/2 \simeq 0.366$, we get the ratio 1.577. □

Corollary 16. *The minimum p.r.c. cost* 2*-star of the input graph is a* 1.577*-approximation solution of the* $\Delta PROCT$ *problem.*

4. An efficient approximation algorithm for the MRCT problem

In this section, an efficient 1.577-approximation algorithm for the MRCT problem shall be presented. Since the MRCT problem is identical to the PROCT problem with all vertex weights equal, by Lemma 15 and Corollary 9, we have the following corollary:

Corollary 17. *If there is an* $O(f(n))$ *time algorithm for finding the minimum routing cost* 2*-star of a metric graph, then the MRCT problem can be approximated with ratio* 1.577 *in* $O(f(n)+n^3)$ *time.*

In [7], there is an $O(n^{2k})$ time algorithm for finding the minimum routing cost k-star of a metric graph. Consequently it leads to an $O(n^4)$ time 1.577-approximation algorithm for the MRCT problem. We shall show that the time complexity can be reduced to $O(n^3 \log n)$ by observing the following property:

Lemma 18. *Let* $T = 2star(x, y, X, Y)$ *be the minimum routing cost* 2*-star of a metric graph* $G = (V, E, w)$. *For any* $u \in X$ *and* $v \in Y$, $w(x,u) - w(y,u) \leqslant w(x,v) - w(y,v)$.

Proof. Similar to Lemma 10, the routing cost of the 2-star T can be computed by the following formula:

$$C(T) = 2|V(X)||V(Y)|w(x,y) + 2(n-1)\left(\sum_{v\in X} w(x,v) + \sum_{v\in Y} w(y,v)\right).$$

When $|V(X)|$ and $|V(Y)|$ are fixed, $C(T)$ depends only on $\sum_{v\in X} w(x,v)$ and $\sum_{v\in Y} w(y,v)$. If the inequality does not hold, we can move u to Y and v to X and obtain another 2-star with smaller routing cost. □

The next lemma shows the time complexity for finding the minimum routing cost 2-star.

Lemma 19. *The minimum routing cost* 2*-star of a metric graph can be found in* $O(n^3 \log n)$ *time.*

Proof. Assume that $2star(x, y, X, Y)$ is the minimum routing cost 2-star with fixed internal nodes x and y. Define a function $f_{x,y}(v) = w(x,v) - w(y,v)$ for all $v \in V$. By sorting the values of $f_{x,y}(v)$, we relabel the vertices such that $V = \{x, y, 1, 2, \ldots, n-2\}$ and $f_{x,y}(i) \leqslant f_{x,y}(i+1)$ for $i = 1, 2, \ldots, n-3$. By Lemma 18, we have $f_{x,y}(u) \leqslant f_{x,y}(v)$ for any $u \in X$ and $v \in Y$. Therefore, there must exist an integer $k \in \{0 \ldots n-2\}$ such that $X = \{x, 1 \ldots k\}$ and $Y = \{y, k+1 \ldots n-2\}$.

To determine the integer k, we compute the routing costs of the $(n-1)$ 2-stars. Let $g(i)$ denote the routing cost of $2star(x,y,\{x,1\dots i\},\{y,i+1\dots n-2\})$. We have

$$g(0)=2(n-1)\sum_{1\leqslant v\leqslant n-2} w(y,v)+2(n-1)w(x,y)$$

and

$$g(i+1)=g(i)+2(n-1)f_{x,y}(i+1)+2(n-2i-3)w(x,y)$$

for $i=1,2,\dots,n-3$. Thus for specified x and y, the best partition (X,Y) can be determined in $O(n)$ time, plus the (dominating) cost $O(n\log n)$ of the sorting procedure. Consequently, the total time complexity is $O(n^3\log n)$ since there are $O(n^2)$ such pairs of vertices x and y. □

The next corollary directly comes from Corollary 17 and Lemma 19.

Corollary 20. *The MRCT problem can be approximated with ratio* 1.577 *in* $O(n^3\log n)$ *time.*

In the rest of this section, we shall show that it is possible to approximate the MRCT problem with ratio 1.577 in $O(n^3)$ time. Instead of the minimum routing cost 2-star, we find a 2-star with minimum routing cost among a subclass of 2-stars.

Let T be the minimum routing cost spanning tree on a metric graph $G=(V,E,w)$ and q,a,b,M,U_i are defined as in Section 3.3 except that each vertex has weight one. Since $|U_1|\geqslant qn$ and $|U_k|\geqslant qn$, we can choose two vertex sets $A\subset U_1$ and $B\subset U_k$ such that $a\in A$ and $b\in B$ and $|A|=|B|=qn$. For the sake of convenience, we assume qn is an integer. Then we construct two 2-stars $T^*=2star(a,b,V\backslash B,B)$ and $T^{**}=2star(a,b,A,V\backslash A)$. Note that when $a=b$, both the 2-stars degenerates to the same 1-star. Similar to Lemma 14 and Lemma 15, we claim a bound on the routing costs of T^* and T^{**}.

Claim 21. $\min\{C(T^*),C(T^{**})\}\leqslant 1.577C(T)$.

Proof. By Lemma 1 and $|B|=qn$,

$$\begin{aligned}C(T^*)&=2(n-1)\sum_{v\notin B} w(v,a)+2(n-1)\sum_{v\in B} w(v,b)+2|B|(n-|B|)w(a,b)\\&\leqslant 2n\sum_{v\notin B} w(v,a)+2n\sum_{v\in B} w(v,b)+2q(1-q)n^2w(a,b).\end{aligned}$$

Similarly,

$$C(T^{**})\leqslant 2n\sum_{v\in A} w(v,a)+2n\sum_{v\notin A} w(v,b)+2q(1-q)n^2w(a,b).$$

By the triangle inequality, $w(u,v)\leqslant d_T(u,v)$ for any vertices u and v. Therefore, for any vertex $v\in A$, $w(v,a)\leqslant d_T(v,M)$. Similarly, $w(v,b)\leqslant d_T(v,M)$ for any vertex $v\in B$.

For any vertex $v \notin A \cup B$, by the triangle inequality, $w(v,a)+w(v,b) \leqslant 2d_T(v,M)+w(M)$. We have

$$\begin{aligned} C(T^*) + C(T^{**}) &\leqslant 4n \sum_{v \in V} d_T(v,M) + 2n \sum_{v \notin A \cup B} w(M) + 4q(1-q)n^2 w(a,b) \\ &= 4n \sum_{v \in V} d_T(v,M) + 2(1-2q^2)n^2 w(M). \end{aligned}$$

Therefore,

$$\min\{C(T^*), C(T^{**})\} \leqslant 2n \sum_{v \in V} d_T(v,M) + (1-2q^2)n^2 w(M). \tag{4}$$

By Lemma 13,

$$C(T) \geqslant 2(1-q)n \sum_{v \in V} d_T(v,M) + 2q(1-q)n^2 w(M). \tag{5}$$

By Eqs. (4) and (5) and setting $q = 0.366$, the ratio is 1.577. □

The next Claim immediately follows Claim 21 and we omit the proof.

Claim 22. *For a metric graph G, there exists a* 1.577*-approximation solution Y of the MRCT problem on G such that Y is either a* 1*-star or a* 2*-star with* 0.366n *leaves connected to one of its internal nodes.*

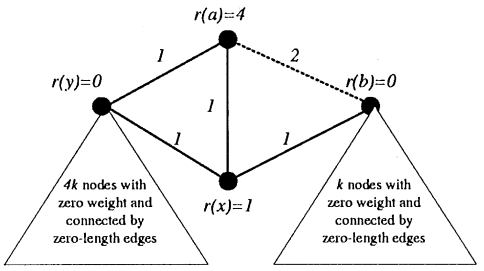
Theorem 23. *The MRCT problem can be approximated with ratio* 1.577 *in* $O(n^3)$ *time.*

Proof. As shown in Corollary 9, we only need to consider the MRCT problem on a metric graph $G = (V,E,w)$. Let Z_1 denote the set of all 1-stars and Z_2 denote the set $\{2star(x,y,V_1,V_2) | x,y \in V, |V_1| = 0.366n\}$. We shall show that the minimum routing cost spanning tree in Z_1 and Z_2 can be found in $O(n^3)$ time. Then the proof is completed by Claim 22.

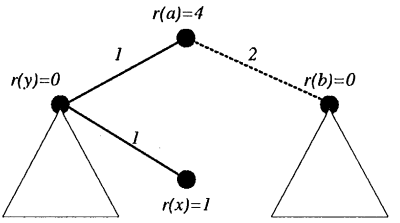
First, the routing costs of all 1-stars can be computed in $O(n^2)$ time since there are n 1-stars whose cost can be computed in $O(n)$ time. The MRCT in Z_2 can be found by an algorithm similar to the one in Lemma 19.

For specified x and y, we first compute $f_{x,y}(v) = w(x,v) - w(y,v)$ for all $v \in V$. Instead of sorting the values of $f_{x,y}(v)$, we find a vertex i such that $f_{x,y}(i)$ is the $(0.366n)$th smallest element in the set $\{f_{x,y}(v) | v \neq x, v \neq y\}$. We then divide V into V_x and V_y such that $|V_x| = 0.366n$ and $f_{x,y}(v) \leqslant f_{x,y}(i)$ for all $v \in V_x$. By Lemma 18, the routing cost of $2star(x,y,V_x,V_y)$ is minimum among the set $\{2star(x,y,V_1,V_2) || V_1| = 0.366n\}$.

Since the kth smallest element among n elements can be found in $O(n)$ time [1] and there are $O(n^2)$ such pairs of x and y, the total time complexity is $O(n^3)$. □



Graph G. (a,b) is not an edge in G



A spanning tree T of the metric closure

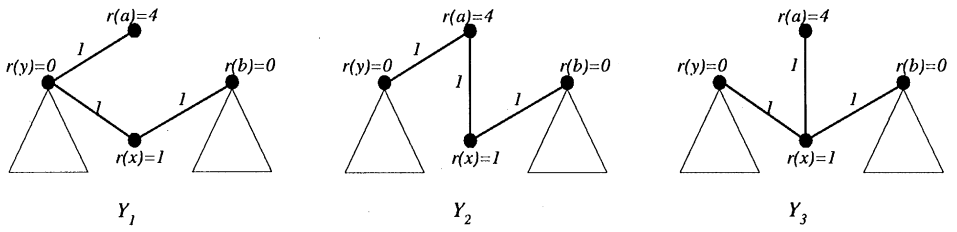


Fig. 5. A tree with bad edges may have less s.r.c. cost.

5. The SROCT problem

For the PROCT problem, it has been shown that the optimal solution for a graph has the same value as the one for its metric closure. In other words, using bad edges cannot lead to a better solution. However, the SROCT problem has no such a property. For example, consider the graph G in Fig. 5. The edge (a,b) is not in $E(G)$, and T is a spanning tree of the metric closure of G. All three possible spanning trees of G are Y_1, Y_2 and Y_3. It can be shown that the s.r.c cost of T is less than that of Y_i for $i = 1,2,3$.

To compare the s.r.c costs, we can only focus on the coefficient of k in the cost. Note that only vertices a and x have nonzero weights. By Lemma 3, the s.r.c. cost of T can be computed as follows:

$$\begin{aligned} C_s(T) &= l_s(T,r,(a,b))w(a,b) + l_s(T,r,(a,y))w(a,y) + l_s(T,r,(y,x))w(x,y) \\ &= 2(k(4+1)+0(4k))2 + 2(k\times 1 + 4\times 4k)(1) + 2(5k\times 1 + 4\times 1)(1) \\ &= 64k + \cdots . \end{aligned}$$

Similarly, we have $C_s(Y_1)=66k$, $C_s(Y_2)=66k$, and $C_s(Y_3)=90k$. The example illustrates that it is impossible to transform any spanning tree of $\bar{G}$ to a spanning tree of G without increasing the s.r.c cost for some graph G, where $\bar{G}$ is the metric closure of G. But it should be noted that the example does not disprove the possibility of reducing the SROCT problem on general graphs to its metric version.

In this section, we shall present a 2-approximation algorithm for the SROCT problem on general graphs. Let G be a graph and $m \in V(G)$. A *shortest-path tree rooted at* m is a spanning tree T of G such that $d_T(v,m) = d_G(v,m)$ for each vertex v. That is, on a shortest-path tree, the path from the root to any vertex is a shortest path on the original graph. The shortest-path tree has been well studied and several efficient algorithms can be found in the literature, e.g. [1,5]. For each vertex v of the input graph, our algorithm finds the shortest-path tree rooted at v. Then it outputs the shortest-path tree with minimum s.r.c. cost. For the approximation ratio, we show that there always exists a vertex m such that the shortest-path tree rooted at m is a 2-approximation solution.

In the following, graph $G=(V,E,w)$ and vertex weight r is the input of the SROCT problem, $|V| = n$, and $R = r(V)$.

Lemma 24. *Let T be a spanning tree of G. For any vertex $m \in V$, $C_s(T) \leqslant 2\sum_{v \in V}(nr(v) + R)d_T(v,m)$.*

Proof.

$$\begin{aligned} C_s(T) &= \sum_{u,v \in V} (r(u) + r(v))d_T(u,v) \\ &\leqslant \sum_{u,v \in V} (r(u) + r(v))(d_T(u,m) + d_T(m,v)) \\ &= 2 \sum_{u,v \in V} (r(u) + r(v))d_T(u,m) \\ &\leqslant 2 \sum_{v \in V} (nr(v) + R)d_T(v,m). \qquad \square \end{aligned}$$

In the following, we use T to denote the optimal spanning tree of the SROCT problem, and use m_1 and m_2 to denote the centroid and r-centroid of T, respectively. Also let $P = SP_T(m_1, m_2)$.

Lemma 25. *For any edge $e \in E(P)$, the s.r.c load $l_s(T,r,e) \geqslant nR$.*

Proof. Let T_1 and T_2 be the two subtrees resulted by deleting e from T. Assume that $m_1 \in V(T_1)$ and $m_2 \in V(T_2)$. By the definitions of centroid and r-centroid, $|V(T_1)| \geqslant n/2$ and $r(T_2) \geqslant R/2$. Then,

$$\begin{aligned} l_s(T,r,e)/2 &= |V(T_1)|r(T_2) + |V(T_2)|r(T_1) \\ &= |V(T_1)|r(T_2) + (n - |V(T_1)|)(R - r(T_2)) \\ &= 2(|V(T_1)| - n/2)(r(T_2) - R/2) + nR/2 \geqslant nR/2. \qquad \square \end{aligned}$$

The next lemma shows a lower bound on the minimum s.r.c. cost. Remind that $d_T(v,P)$ denotes the shortest path length from a vertex v to path P.

Lemma 26. $C_s(T) \geqslant \sum_{v \in V}(nr(v)+R)d_T(v,P) + nRw(P)$.

Proof. For any vertex u, we define $SB(u)=\{v|SP_T(u,v)\cap P=\emptyset\}$. Note that $|SB(u)| \leqslant n/2$ and $r(SB(u)) \leqslant R/2$ for any vertex u by the definitions of centroid and r-centroid:

$$\begin{aligned} C_s(T) &= \sum_{u,v \in V}(r(u)+r(v))d_T(u,v) \\ &= 2\sum_{u,v \in V} r(u)d_T(u,v) \\ &\geqslant 2\sum_{u \in V}\sum_{v \notin SB(u)} r(u)(d_T(u,P)+d_T(v,P)) \\ &\quad +2\sum_{u,v \in V} r(u)w(SP_T(u,v)\cap P). \end{aligned} \tag{6}$$

For the first term in Eq. (6),

$$\begin{aligned} &2\sum_{u \in V}\sum_{v \notin SB(u)} r(u)(d_T(u,P)+d_T(v,P)) \\ &=2\sum_{u \in V}\sum_{v \notin SB(u)} r(u)d_T(u,P) + 2\sum_{u \in V}\sum_{v \notin SB(u)} r(u)d_T(v,P) \\ &\geqslant \sum_{u \in V} nr(u)d_T(u,P) + 2\sum_{v \in V}\sum_{u \notin SB(v)} r(u)d_T(v,P) \\ &\geqslant \sum_{u \in V} nr(u)d_T(u,P) + \sum_{v \in V} Rd_T(v,P) \\ &= \sum_{v \in V}(nr(v)+R)d_T(v,P). \end{aligned} \tag{7}$$

For the second term in Eq. (6),

$$\begin{aligned} &2\sum_{u,v \in V} r(u)w(SP_T(u,v)\cap P) \\ &=2\sum_{u,v \in V} r(u)\left(\sum_{e \in SP_T(u,v)\cap P} w(e)\right) \\ &= \sum_{e \in E(P)}\left(2\sum_{v} r(\{u|e \in E(SP_T(u,v))\})\right)w(e) \\ &= \sum_{e \in E(P)} l_s(T,r,e)w(e) \\ &\geqslant nRw(P) \quad \text{(by Lemma 25).} \end{aligned} \tag{8}$$

By Eqs. (6)–(8), the proof is completed. □

The main result of this section is stated in the next theorem.

Theorem 27. *There exists a 2-approximation algorithm with time complexity* $O(n^3)$ *for the SROCT problem.*

Proof. Let Y^* and Y^{**} be the shortest-path trees rooted at m_1 and m_2, respectively. Also, for any $v \in V$, let $h_1(v) = w(SP_T(v, m_1) \cap P)$ and $h_2(v) = w(SP_T(v, m_2) \cap P)$. By Lemma 24,

$$\begin{aligned} C_s(Y^*)/2 &\leqslant \sum_{v \in V} (nr(v) + R) d_{Y^*}(v, m_1) \\ &\leqslant \sum_{v \in V} (nr(v) + R)(d_T(v, P) + h_1(v)). \end{aligned} \tag{9}$$

Similarly,

$$C_s(Y^{**})/2 \leqslant \sum_{v \in V} (nr(v) + R)(d_T(v, P) + h_2(v)). \tag{10}$$

Since $h_1(v) + h_2(v) = w(P)$ for any vertex v, by Eqs. (9) and (10), we have

$$\begin{aligned} &\min\{C_s(Y^*), C_s(Y^{**})\} \\ &\quad \leqslant (C_s(Y^*) + C_s(Y^{**}))/2 \\ &\quad \leqslant \sum_{v \in V} (nr(v) + R)(2d_T(v, P) + h_1(v) + h_2(v)) \\ &\quad = \sum_{v \in V} (nr(v) + R)(2d_T(v, P) + w(P)) \\ &\quad = 2 \sum_{v \in V} (nr(v) + R) d_T(v, P) + 2nRw(P) \\ &\quad \leqslant 2C_s(T) \quad \text{(by Lemma 26).} \end{aligned}$$

We have proved that there exists a vertex m such that any shortest-path tree rooted at m is a 2-approximation solution. Since it takes $O(n^2)$ time to construct a shortest-path tree rooted at a given vertex and the s.r.c cost of a tree can be computed in $O(n)$ time, by computing the shortest-path tree rooted at each vertex and choosing the one with minimum s.r.c cost, a 2-approximation solution of the SROCT problem can be found in $O(n^3)$ time. □

6. Concluding remarks

As mentioned in Section 1, it can be shown that the PROCT problem can be approximated arbitrarily close to 1 by the minimum p.r.c. cost k-star when k is sufficiently large. Therefore, any polynomial time algorithm for the minimum p.r.c. cost k-star will result in a PTAS for the PROCT problem. However, it is still unknown how to construct the minimum k-star in polynomial time when $k \geqslant 3$. Another interesting related problem is the Steiner MRCT problem, in which we want to minimize the total path length summed over all pairs of vertices in a given vertex subset. The Steiner MRCT

problem is a special case of the PROCT problem, in which the vertex weights are either 0 or 1. It is not hard to generalize the PTAS for the MRCT problem to a PTAS for the Steiner MRCT problem.

For the MRCT problem, an interesting question is how to efficiently find the minimum routing cost k-star for any fixed k. This will improve the time complexity of the PTAS in [7]. In this paper, for $k = 2$, we have shown an algorithm which is more time efficient than the one in [7]. But we did not find a way to generalize the idea to obtain a more time efficient algorithm for the case $k > 2$.

For the SROCT problem, the result in this paper is only the first attempt to its approximability. Future work includes improving the approximation ratio for both general and metric inputs.

Acknowledgements

We thank the anonymous referees for their careful reading and many useful comments.

References

[1] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, MIT Press, Cambridge, MA, 1994.
[2] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Francisco, 1979.
[3] T.C. Hu, Optimum communication spanning trees, SIAM J. Comput. 3(3) (1974) 188–195.
[4] D.S. Johnson, J.K. Lenstra, A.H.G. Rinnooy Kan, The complexity of the network design problem, Networks 8 (1978) 279–285.
[5] R.E. Tarjan, Data Structures and Network Algorithms, SIAM Press, Philadelphia, PA, 1983.
[6] R. Wong, Worst-case analysis of network design problem heuristics., SIAM J. Algebraic Discrete Math. 1 (1980) 51–63.
[7] B.Y. Wu, G. Lancia, V. Bafna, K.M. Chao, R. Ravi, C.Y. Tang, A polynomial time approximation scheme for minimum routing cost spanning trees, SIAM J. Comput. in press. (A preliminary version of this paper appears in the Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'98), 1998, pp. 21–32.)