A note on a polynomial time approximation scheme for the MRCT problem

Bang Ye Wu Kun–Mao Chao

April 1, 2004

1 Further improvement

Let S be a minimal δ-separator of $\widehat{T}$. The strategy of algorithms shown in our previous note on 3/2-approximation is to "guess" the structure of S and to construct a general star with the guessed structure as the core. If $T \in \text{star}(S)$, by the lemmas in the previous note,

$$C(T) \leq 2n \sum_{v \in V(G)} d_G(v, S) + (n^2/2)w(S),$$

and

$$C(\widehat{T}) \geq 2(1-\delta)n \sum_{v \in V} d_{\widehat{T}}(v, S) + 2\delta(1-\delta)n^2 w(S).$$

The approximation ratio, by comparing the two inequalities, is

$$\max\{\frac{1}{1-\delta}, \frac{1}{4\delta(1-\delta)}\}.$$

The ratio achieves its minimum when the two terms coincide, i.e., $\delta = 1/4$, and the minimum ratio is 4/3. In fact, by using a general star and a (1/4)-separator, it is possible to approximate an MRCT with ratio $(4/3) + \varepsilon$ for any constant $\varepsilon > 0$ in polynomial time. The additional error ε is due to the difference between the guessed and the true separators.

By this strategy, the approximation ratio is limited even if S was known exactly. The limit of the approximation ratio may be mostly due to that we consider only general stars. In a general star, the vertices are always connected to their closest vertices of the core. In extreme cases, roughly half of the vertices connected are to both sides of a costly edge. This results in the cost $(n^2/2)w(S)$ in the upper bound of a general star. To make a

breakthrough, the restriction that each vertex is connected to the closest vertex of the core needs to be relaxed.

A metric graph is a complete graph with triangle inequality, i.e., each edge is a shortest path of its two endpoints. If the input graph is a metric graph, the core will be a two-edge path, and each vertex is adjacent to one of the "critical vertices"— a centroid and the two endpoints of a path separator. Define k-stars to be the trees with at most k internal vertices. The constructed approximation solution is a 3-star. More importantly, k-stars have no such restriction like general stars and can be used to approximate an MRCT more precisely. Later in this chapter we shall see how it works.

However, k-stars work only for metric graphs. The class of metric graphs is an important subclass of graphs. Solving the MRCT problem on metric graphs is itself meaningful. Before considering the approximation problem on metric graphs, two questions come to our minds:

- What is the computational complexity of the MRCT problem on metric graphs, NP-hard or polynomial-time solvable?
- If it is NP-hard, does its approximability differ from that of the general problem?

We shall answer the questions in the next section.

2 A Reduction to the Metric Case

In this section, we shall show that the MRCT problem on general inputs can be reduced to the same problem with metric inputs. The reduction is done by a transformation algorithm.

Definition 1: The metric closure of a graph $G = (V, E, w)$ is the complete graph $\bar{G} = (V, V \times V, \bar{w})$ in which $\bar{w}(u, v) = d_G(u, v)$ for all $u, v \in V$.

Let $G = (V, E, w)$ and $\bar{G} = (V, V \times V, \bar{w})$ be its metric closure. Any edge (a, b) in $\bar{G}$ is called a *bad edge* if $(a, b) \notin E$ or $w(a, b) > \bar{w}(a, b)$. For any bad edge $e = (a, b)$, there must exist a path $P = SP_G(a, b) \neq e$ such that $w(P) = \bar{w}(a, b)$. Given any spanning tree T of $\bar{G}$, the algorithm can construct another spanning tree Y without any bad edge such that $C(Y) \leq C(T)$. Since Y has no bad edge, $\bar{w}(e) = w(e)$ for all $e \in E(Y)$, and Y can be thought of as a spanning tree of G with the same routing cost. The algorithm is listed in the following.

Algorithm: REMOVE_BAD
Input: A spanning tree T of $\bar{G}$.
Output: A spanning tree Y of G such that $C(Y) \leq C(T)$.

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 $SP_G(a, b) = (a, x, ..., b)$ and y is the parent of x */
if b is not an ancestor of x **then**
$Y^* \leftarrow T \cup (x, b) - (a, b)$;$Y^{**} \leftarrow Y^* \cup \{(a, x)\} - \{(x, y)\}$;
else
$Y^* \leftarrow T \cup (a, x) - (a, b)$;$Y^{**} \leftarrow Y^* \cup \{(b, x)\} - \{(x, y)\}$;
if $C(Y^*) < C(Y^{**})$ **then** $Y \leftarrow Y^*$ **else** $Y \leftarrow Y^{**}$
(II) $T \leftarrow Y$

The algorithm computes Y by iteratively replacing the bad edges until there is no bad edge. It will be shown that the cost is never increased at each iteration and it takes no more than $O(n^2)$ iterations. We assume that the shortest paths obtained in the first step have the following property: If $SP_G(a, b) = (a, x, \ldots, b)$, then $SP_G(a, b) = (a, x) \cup SP_G(x, b)$. This assumption is not strong since almost all popular algorithms for all-pairs shortest paths output such a solution.

Proposition 1: The while loop in Algorithm REMOVE_BAD is executed at most $O(n^2)$ times.

Proof: For each bad edge $e = (a, b)$, let $h(e)$ be the number of edges in $SP_G(a, b)$ and $f(T) = \sum_{\text{bad } e} h(e)$. Since $h(e) \leq n - 1$, $f(T) < n^2$ initially. Since (a, x) is not a bad edge, it is easy to check that $f(T)$ decreases by at least 1 at each iteration. □

Proposition 2: Before instruction **(II)** is executed, $C(Y) \leq C(T)$.

Proof: For any node v, let $S_v = V(T_v)$. As shown in Figure 1, there are two cases. Case 2 is identical to Case 1 if the tree is re-rooted at b and the roles of a and b are exchanged. Therefore, only the inequality for Case 1 needs to be proved, i.e., $x \in S_a - S_b$.

If $C(Y^*) \leq C(T)$, the result follows. Otherwise, let $U_1 = S_a - S_b$ and $U_2 = S_a - S_b - S_x$. Since the distance does not change for any two vertices

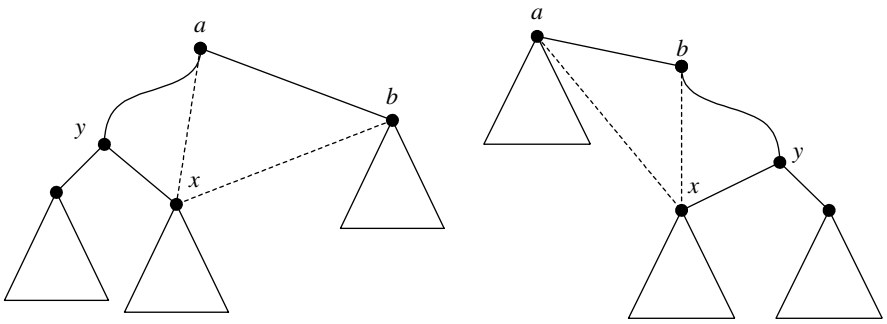

Figure 1: Remove bad edge (a, b). Case 1 (left) and Case 2 (right).

both in U_1 (or both in S_b), we have

$$\begin{aligned} C(T) &< C(Y^*) \\ \Rightarrow \sum_{u\in U_1}\sum_{v\in S_b} d_T(u,v) &< \sum_{u\in U_1}\sum_{v\in S_b} d_{Y^*}(u,v). \end{aligned}$$

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

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

Note that $S_b \neq \emptyset$ since the inequality holds. By the definition of the metric closure, $\bar{w}(a,b) = \bar{w}(a,x) + \bar{w}(x,b)$, and then

$$\sum_{u\in U_1} (d_T(u,a) - d_T(u,x)) < -|U_1|\bar{w}(a,x). \tag{1}$$

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

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

Since $d_{Y^{**}}(u,v) \leq d_T(u,v)$ 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 $d_{Y^{**}}(u,v) = d_T(u,a) + \bar{w}(a,x) + d_T(x,v)$ for any $u \in U_2$ and $v \in S_x$, we obtain

$$\begin{aligned}& (C(Y^{**}) - C(T))/2 \\ \leq\ & \sum_{u\in U_2}\sum_{v\in S_x}(d_T(u,a) + \bar{w}(a,x) - d_T(u,x)) \\ =\ & |S_x|\sum_{u\in U_2}(d_T(u,a) + \bar{w}(a,x) - d_T(u,x)) \\ =\ & |S_x|\sum_{u\in U_2}(d_T(u,a) - d_T(u,x)) + |U_2||S_x|\bar{w}(a,x) \\ \leq\ & |S_x|\sum_{u\in U_1}(d_T(u,a) - d_T(u,x)) + |U_2||S_x|\bar{w}(a,x) \qquad (2) \\ <\ & -|U_1||S_x|\bar{w}(a,x) + |U_2||S_x|\bar{w}(a,x) \qquad (3) \\ \leq\ & 0.\end{aligned}$$

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

The next lemma follows Propositions 1 and 2, and that each iteration can be done in $O(n)$ time.

Lemma 1: For any spanning tree $\bar{T}$ of $\bar{G}$, it can be transformed into a spanning tree T of G in $O(n^3)$ time and $C(T) \leq C(\bar{T})$.

The above lemma implies that $C(\text{mrct}(G)) \leq C(\text{mrct}(\bar{G}))$. It is easy to see that $C(\text{mrct}(G)) \geq C(\text{mrct}(\bar{G}))$. Therefore, we have the following corollary.

Corollary 2: $C(\text{mrct}(G)) = C(\text{mrct}(\bar{G}))$.

Let ΔMRCT denote the MRCT problem with metric inputs. We have the next theorem.

Theorem 3: If there is a $(1+\varepsilon)$-approximation algorithm for ΔMRCT with time complexity $O(f(n))$, then there is a $(1+\varepsilon)$-approximation algorithm for MRCT with time complexity $O(f(n)+n^3)$.

Proof: Let G be the input graph for the MRCT problem. The metric closure $\bar{G}$ can be constructed in time $O(n^2 \log n + mn)$. If there is a $(1+\varepsilon)$-approximation algorithm for the ΔMRCT problem, a spanning tree T of $\bar{G}$ can be computed in time $O(f(n))$ such that $C(T) \leq (1+\varepsilon)C(\text{mrct}(\bar{G}))$. Using Algorithm REMOVE_BAD, a spanning tree Y of G can be constructed such that $C(Y) \leq C(T) \leq (1+\varepsilon)C(\text{mrct}(\bar{G}))= (1+\varepsilon)C(\text{mrct}(G))$. The overall time complexity is then $O(f(n)+n^3)$. □

Corollary 4: The ΔMRCT problem is NP-hard.

3 A Polynomial Time Approximation Scheme

3.1 Overview

We sketch a *Polynomial Time Approximation Scheme* (PTAS) for the MRCT problem in this section.

As described previously, the fact that the costs w may not obey the triangle inequality is irrelevant, since we can simply replace these costs by their metric closure. Therefore, in this section we may assume that $G = (V, E, w)$ is a metric graph.

We use k-stars, i.e., trees with no more than k internal nodes, as a basis of our approximation scheme. In Section **??** we show that for any constant k, a minimum routing cost k-star can be determined in polynomial time. In order to show that a k-star achieves a $(1+\varepsilon)$ approximation, we show that, for any tree T and constant $\delta \leq 1/2$:

1. It is possible to determine a δ-separator, and the separator can be cut into several δ-paths such that the total number of cut nodes and leaves of the separator is at most $\lceil \frac{2}{\delta} \rceil - 3$. (Lemma **??**)

2. Using the separator, T can be converted into a $(\lceil \frac{2}{\delta} \rceil - 3)$-star X, whose internal nodes are just those cut nodes and leaves. The routing cost of X satisfies $C(X) \leq (1 + \frac{\delta}{1-\delta})C(T)$. (Lemma **??**)

By using $T = \widehat{T}$ =mrct(G), $\delta = \frac{\varepsilon}{1+\varepsilon}$ and finding the best $(\lceil \frac{2}{\delta} \rceil - 3)$-star K, we obtain the desired approximation

$$C(K) \leq (1 + \frac{\delta}{1-\delta})C(\widehat{T}) = (1+\varepsilon)C(\widehat{T}).$$

Before going into the details of the general case, take a look at how to find a 3/2-approximation of an MRCT and its performance analysis.

Recall that a centroid of a tree is the vertex whose removal cuts the tree into components of no more than $n/2$ vertices. Let $\widehat{T}$ be an MRCT of a metric graph G. Root $\widehat{T}$ at its centroid r. For an edge (u, v) with parent v, the routing load of the edge is $2x(n-x)$, in which x is the number of descendants of u. (For convenience, we assume that a vertex is also a descendant of itself.) For a desired positive $\delta \leq 1/2$, removing all vertices with number of descendants no more than δn, we may obtain a connected subgraph S of T. The subgraph S is a minimal δ-separator. In the case that $\delta = 0.5$, S contains only the centroid. If the δ-separator S of the MRCT is given and, for each vertex not in S, its lowest ancestor in S is also known, we may easily construct a $1/(1-\delta)$-approximation Y of the MRCT as follows:

- $S \subset Y$.
- For each vertex u not in S, connect it to its lowest ancestor v in S by adding edge (u, v).

The approximation ratio of Y can be shown by the following two observations.

- For each edge in S, the routing load in Y is the same as that in $\widehat{T}$.
- For each edge (u, v) not in S, the routing load in Y is $2(n-1)$. However, there is a path from u to v in $\widehat{T}$ of which each edge has routing load no less than $2(1-\delta)n$, and the length of the path is at least the same as the edge (u, v).

For example, consider the simplest case that $\delta = 1/2$. A 1/2-separator contains only one vertex. Assume that r be such a vertex on an MRCT $\widehat{T}$. For every other vertex, the ancestor in the separator is r. Connecting all other nodes to r, we obtain a star Y. The routing cost of Y is the sum of all edges (v, r) multiplied by its routing load $2(n-1)$. For each vertex v, there is a path from v to r in $\widehat{T}$. Since r is a 1/2-separator of $\widehat{T}$, the routing load of the path is at least n and the path is no shorter than the edge (v, r) by the triangle inequality. Consequently Y is a 2-approximation of the MRCT $\widehat{T}$. Since the separator contains only one vertex, we may try all possible vertices and it leads to an $O(n^2)$ time 2-approximation of the MRCT problem on metric graphs.

But when $\delta < 1/2$, the separator may contain as many as $\Omega(n)$ vertices, and it is too costly to enumerate all possible separators. However, instead of

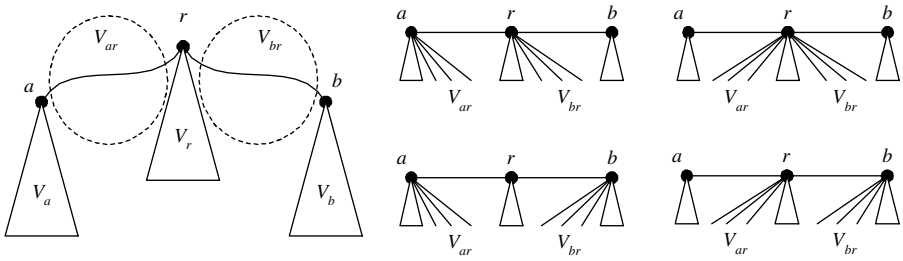

Figure 2: An MRCT and the four 3-stars.

the entire separator, we may achieve the same ratio by knowing only some *critical* vertices of the separator. In the following, we shall take $\delta = 1/3$ as an example to show that we only need to know three critical vertices of the separator to construct a 3/2-approximation.

Root the MRCT $\widehat{T}$ at its centroid r. There are at most two subtrees which contain more than $n/3$ nodes. Let a and b be the lowest vertices with at least $n/3$ descendants. We ignore the cases where $r = a$ or $r = b$. For such cases, the ratio can be shown similarly. Let P be the path from a to b. Obviously the path contains r and is a minimal 1/3-separator. We shall transform T into a 3-star with internal nodes a, b, and r such that its routing cost is no more than 3/2 of that of $\widehat{T}$. As shown in Figure 2, let us partition all the vertices into V_a, V_b, V_r, V_{ar}, and V_{br}. The sets V_a, V_b, and V_r contain the vertices whose lowest ancestor on P is a, b, and r, respectively. The set V_{ar} (and V_{br}) consists of the vertices whose lowest ancestor on P is between a and r (and between b and r, respectively).

First replace P with edges (a, r) and (b, r). For each vertex v in V_a (or V_b, V_r), edge (v, a) (or (v, b), (v, r) respectively) is added. For the vertices in V_{ar}, we consider two cases. Either all of them are connected to a or all of them are connected to r. The vertices in V_{br} are connected similarly. The four possible 3-stars are illustrated in Figure 2.

Now consider the routing cost of $\widehat{T}$. For each vertex v, the routing load of the path from v to P is no less than $4n/3$ since P is a 1/3-separator. For each edge e of P, since there are at least $n/3$ nodes on either side of it, the routing load is no less than $2(n/3)(2n/3) = 4n^2/9$. Therefore we have the following lower bound of the routing cost of the MRCT:

$$C(\widehat{T}) \geq (4n/3) \sum_v d_{\widehat{T}}(v, P) + (4/9)n^2 w(P).$$

In either of the 3-stars constructed above, for each vertex v in $V_a \cup V_r \cup V_b$, the routing load of the edge incident to v is $2(n-1)$, and the edge length is at most the same as the path from v to P on T. For each node v in V_{ar}, by the triangle inequality, we have

$$(w(v,a) + w(v,r))/2 \leq d_{\widehat{T}}(v,P) + d_{\widehat{T}}(a,r)/2.$$

Note that there are no more than $n/6$ nodes in V_{ar}. For edge (a,r), the routing load is no more than $2(n/2)(n/2) = n^2/2$. (Note: For this simple case that $\delta = 1/3$, this bound is enough. However, a more precise analysis of the incremental routing load is required for smaller δ.)

For the nodes in V_{br}, we may obtain a similar result. In summary, the minimum routing cost of the constructed 3-stars is no more than

$$\begin{aligned} & 2(n-1)\sum_v d_{\widehat{T}}(v,P) + (n^2/6)(d_{\widehat{T}}(a,r) + d_{\widehat{T}}(b,r)) + (1/2)n^2 w(P) \\ \leq \; & 2n\sum_v d_{\widehat{T}}(v,P) + (2/3)n^2 w(P). \end{aligned}$$

The approximation ratio, by comparing with the lower bound of the optimal, is 3/2.

The method can be extended to any $\delta \leq 0.5$. Let S be a δ-separator of T. The critical vertex set, defined as the *cut and leaf set* in the next section, to construct a $1/(1-\delta)$-approximation k-star consists of the following vertices.

- The leaves of S as a and b in the above example.
- The vertices with more than two neighbors on S.
- Some additional vertices such that all the critical vertices cut the separator into edge-disjoint paths and the number of vertices whose lowest ancestors on S belong to the same path is no more than $\delta n/2$. Such a path is defined as a δ-path in the next section. In the above example, r is such a vertex. The vertices a, b and r cut the separator into two paths, and the number of vertices in either V_{ar} or V_{br} is no more than $n/6$.

We shall show that the number of the necessary critical vertices is at most $2/\delta - 3$ for any $\delta \leq 0.5$. Consequently there exists a $(2/\delta - 3)$-star which is an approximation of an MRCT with ratio $1/(1-\delta)$. The PTAS is to construct the $(2/\delta - 3)$-star of minimum routing cost.

The core of a tree is the subgraph obtained by removing all its leaves. The core of a k-star contains no more than k vertices and therefore the

number of all possible cores is polynomial to k. For each possible core, the algorithm finds the best way to connect the leaves to one of the vertices of the core. A k-component integer vector $(n_1, n_2, \ldots, n_k)$ is used to indicate how many leaves will be connected to each of the k vertices of the core, in which $\sum_i n_i = n - k$. There are $O(n^{k-1})$ such vectors. For each core and each vector, the routing load on each core edge is fixed since the number of vertices on both sides of the edge are specified by the core and the vector. Therefore the best leaf connection is determined by the leaf edges subject to the numbers of leaves to be connected to the vertices of the core. Such a problem can be solved by solving an assignment problem in $O(n^3)$ time. Consequently the minimum routing cost k-star can be constructed in time polynomial to k and n.

Consider an example for $k = 3$. For a 3-star, the core is a 3-vertex path. There are $O(n^3)$ possible cores. For each possible core (a, b, c), use a three-component integer vector (x, y, z) to indicate how many leaves will be connected to a, b, and c, in which $x+y+z = n-3$ and x, y, z are nonnegative integers. There are $O(n^2)$ such vectors. For a specified core and a specified vector, the routing load on each core edge is also fixed. That is, the routing load on (a, b) is $2(x + 1)(n - x - 1)$ and on (b, c) is $2(z + 1)(n - z - 1)$. Therefore the best leaf connection is determined by the leaf edges subject to the numbers of leaves to be connected to a, b, c. Such a problem can be solved by solving an assignment problem in $O(n^3)$ time. The total time complexity is $O(n^{2k+2})$, which is polynomial for constant k. In fact we need not solve the individual assignment problem for the best leaf connection of each vector. The best leaf connection of one vector can be found from that of another vector by solving a shortest path problem if the two vectors are adjacent. Two vectors are adjacent if one can be obtained from the other by increasing a component by one and decreasing a component by one, e.g., $(5, 4, 2)$ and $(5, 3, 3)$. With this result, the time complexity is reduced to $O(n^{2k})$.