A note on 15/8 & 3/2-approximation algorithms for the MRCT problem

Bang Ye Wu Kun–Mao Chao

March 18, 2004

1 Approximating by a General Star

1.1 Separators and general stars

A key point to the 2-approximation in our previous note is the existence of the centroid, which separates a tree into sufficiently small components. To generalize the idea, we define the separator of a tree in Definition 1.

Definition 1: Let T be a spanning tree of G and S be a connected subgraph of T. A *branch* of S is a connected component of the subgraph that results by removing S from T.

Definition 2: Let $\delta \leq 1/2$. A connected subgraph S is a δ-separator of T if $|B| \leq \delta|V(T)|$ for every branch B of S.

A δ-separator S is *minimal* if any proper subgraph of S is not a δ-separator of T.

Example 1: The tree in Figure 1(a) has 26 vertices in which v_1 is a centroid. The vertex v_1 is a minimal 1/2-separator. As shown in (b), each branch contains no more than 13 vertices. But v_1, or even the edge (v_1, v_2), is not a 1/3-separator because there exists a subtree whose number of vertices is nine, which is greater than 26/3. The path between v_2 and v_3 is a minimal 1/3-separator (Frame (c)), and the subgraph that consists of v_1, v_2, v_3, v_4, and v_5 is a minimal 1/4-separator (Frame (d)).

The δ-separator can be thought of as a generalization of the centroid of a tree. Obviously, a centroid is a 1/2-separator which contains only one node. Intuitively, a separator is like a routing center of the tree. Starting from

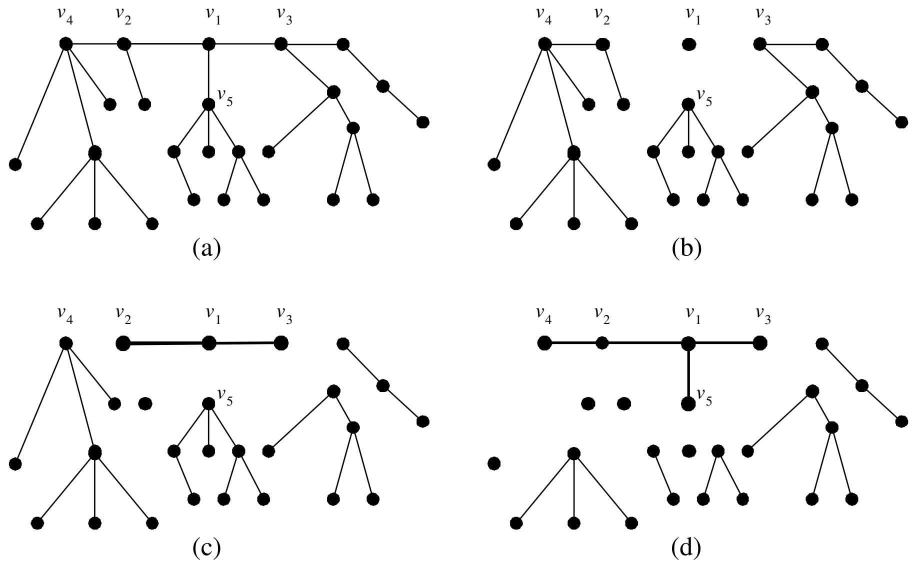

Figure 1: An example of a minimal separator of a tree.

any node, there are sufficiently many nodes which can only be reached after reaching the separator. For two vertices i and j in different components separated by S, the path between them can be divided into three subpaths: from i to S, a path in S, and from S to j. Since each component contains no more than δn vertices, the distance $d_T(i, S)$ will be counted at least $2(1-\delta)n$ times as we compute the routing cost of T. For each edge e in S, since there are at least δn vertices on either side of the edge, by Fact **??**, the routing load on e is at least $2\delta(1-\delta)n^2$. Some notations are given below and illustrated in Figure 2.

Definition 3: Let T be a spanning tree of G and S be a connected subgraph of T. Let u be a vertex in S. The set of branches of S connected to u by an edge of T is denoted by $brn(T, S, u)$, while $brn(T, S)$ is for the set of all branches of S. The set of vertices in the branches connected to u is denoted by $VB(T, S, u) = \{u\} \cup \{v | v \in B \in brn(T, S, u)\}$.

The next fact directly follows the definitions.

Fact 1: Let S be a minimal δ-separator of T. If v is a leaf of S, then $|VB(T, S, v)| > \delta |V(T)|$.

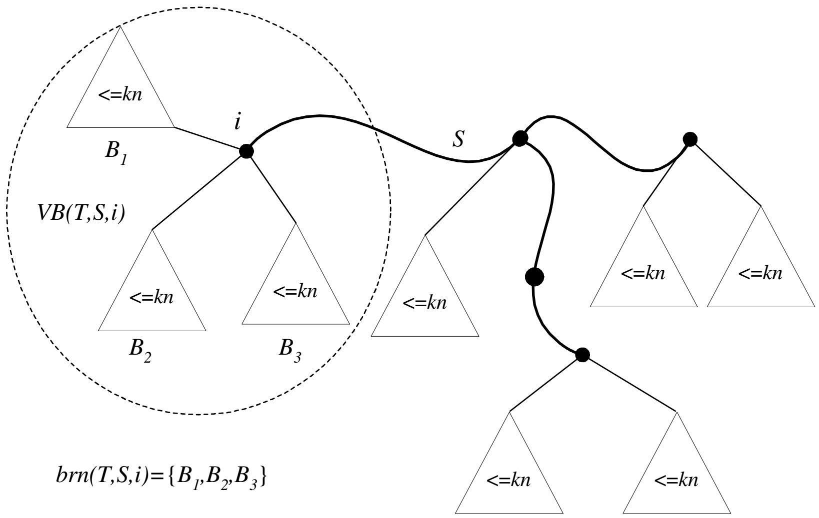

Figure 2: A δ-separator and branches of a tree. The bold line is the separator S and each triangle is a branch of S.

A star is a tree with only one internal vertex (center). We define a *general star* as follows.

Definition 4: Let R be a tree contained in the underlying graph G. A spanning tree T is a general star with core R if each vertex is connected to R by a shortest path.

For an extreme example, a shortest-paths tree is a general star whose core contains only one vertex. By star(R), we denote the set of all general stars with core R. The intuition of using general stars to approximate an MRCT is described as follows: Assume S is a δ-separator of an optimal tree T. The separator breaks the tree into sufficiently small components (branches). The routing cost of T is the sum of the distances of the $n(n-1)$ pairs of vertices. If we divide the routing cost into two terms, the total distance of vertices in different branches and the total distance of vertices in a same branch, then the inter-branch distance is the larger fraction of the total routing cost. Furthermore, the fraction gets larger and larger when a smaller δ is chosen. If we construct a general star with core S, the routing cost will be very close to the optimal.

Given a core, to construct a general star is just to find a shortest-paths forest, which can be done in $O(n \log n + m)$ time. However, it can be done more efficiently if the all-pairs shortest paths are given.

Lemma 1: Let G be a graph, and let S be a tree contained in G. A spanning tree $T \in \text{star}(S)$ can be found in $O(n)$ time if a shortest path $SP_G(v, S)$ is given for every $v \in V(G)$.

Proof: A constructive proof is given below. Starting from $T = S$, we show a procedure which inserts all other vertices into T one by one. At each iteration, the following equality is kept:

$$d_T(v, S) = d_G(v, S) \qquad \forall v \in V(T). \tag{1}$$

It is easy to see that (1) is true initially. Consider the step of inserting a vertex. Let $SP_G(v, S) = (v = v_1, v_2, ..., v_k \in S)$ be a shortest path from v to S, and let v_j be the first vertex which is already in T. Set $T \leftarrow T \cup (v_1, v_2, ..., v_j)$. Since $(v_1, v_2, ..., v_k)$ is a shortest path from v to S, $(v_a, v_{a+1}, ..., v_j)$ is also a shortest path from v_a to v_j for any $a = 1, \dots j$, and (1) is true. It is easy to see that the time complexity is $O(n)$, if a shortest path from v to S is given for every $v \in V$. □

Let S be a connected subgraph of a spanning tree T. The path between two vertices v and u in different branches can be divided into three subpaths: the path from v to S, the path contained in S, and the path from u to S. For convenience, we define $d_T^S(u, v) = w(SP_T(u, v) \cap S)$. Obviously

$$d_T(u, v) \leq d_T(v, S) + d_T^S(u, v) + d_T(u, S), \tag{2}$$

and the equality holds if v and u are in different branches. Summing up (2) for all pairs of vertices, we have

$$C(T) \leq 2n \sum_{v \in V} d_T(v, S) + \sum_{u,v \in V} d_T^S(u, v).$$

By the definition of routing load,

$$\sum_{u,v \in V} d_T^S(u, v) = \sum_{e \in E(S)} l(T, e) w(e).$$

Suppose that T is a general star with core S. We can establish an upper bound of the routing cost by observing that $d_T(v, S) = d_G(v, S)$ for any vertex v and $l(T, e) \leq \frac{n^2}{2}$ for any edge e (Fact **??**).

Lemma 2: Let G be a graph and S be a tree contained in G. If $T \in \text{star}(S)$, $C(T) \leq 2n \sum_{v \in V(G)} d_G(v, S) + (n^2/2) w(S)$.

Now we establish a lower bound of the minimum routing cost. Let S be a minimal δ-separator of a spanning tree T and $\mathcal{X}$ denote the set of the ordered pairs of the vertices not in a same branch of S. For any vertex pair $(u,v) \in \mathcal{X}$,

$$d_T(u,v) = d_T(u,S) + d_T^S(u,v) + d_T(v,S). \tag{3}$$

Summing up (3) for all pairs in $\mathcal{X}$, we have a lower bound of $C(T)$.

$$\begin{aligned} C(T) &\geq \sum_{(u,v)\in\mathcal{X}} d_T(u,v) \\ &= \sum_{(u,v)\in\mathcal{X}} (d_T(u,S) + d_T(v,S)) + \sum_{(u,v)\in\mathcal{X}} d_T^S(u,v). \end{aligned} \tag{4}$$

Since S is a δ-separator, there are at least $(1-\delta)n$ vertices not in the same branch of any vertex v, and we have

$$\sum_{(u,v)\in\mathcal{X}} (d_T(u,S) + d_T(v,S)) \geq 2(1-\delta)n \sum_{v\in V} d_T(v,S). \tag{5}$$

Since $d_T^S(u,v) = 0$ if v and u are in the same branch,

$$\sum_{(u,v)\in\mathcal{X}} d_T^S(u,v) = \sum_v \sum_u d_T^S(u,v).$$

By definition, this is the total routing cost on the edges of S. Rewriting this in terms of routing loads, we have

$$\sum_v \sum_u d_T^S(u,v) = \sum_{e\in E(S)} l(T,e)w(e). \tag{6}$$

Substituting (5) and (6) in (4), we have

$$C(T) \geq 2(1-\delta)n \sum_{v\in V} d_T(v,S) + \sum_{e\in E(S)} l(T,e)w(e). \tag{7}$$

Since S is a minimal δ-separator, for any edge of S there are at least δn vertices on either side of the edge. Therefore, $l(T,e) \geq 2\delta(1-\delta)n^2$ for any $e \in E(S)$. Consequently,

$$\sum_{e\in E(S)} l(T,e)w(e) \geq 2\delta(1-\delta)n^2 \sum_{e\in E(S)} w(e) = 2\delta(1-\delta)n^2 w(S). \tag{8}$$

Combining (7) and (8), we obtain

$$C(T) \geq 2(1-\delta)n \sum_{v\in V} d_T(v,S) + 2\delta(1-\delta)n^2 w(S). \tag{9}$$

Particularly, for the MRCT $\widehat{T}$ we have the next lemma.

Lemma 3: If S is a minimal δ-separator of $\widehat{T}$, then

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

1.2 A 15/8-approximation algorithm

In our previous note, a 1/2-separator is used to derive a 2-approximation algorithm. The idea is now generalized to show that a better approximation ratio can be obtained by using a 1/3-separator. The following lemma shows the existence of a 1/3-separator. Note that a path may contain only one vertex.

Lemma 4: For any tree T, there is a path $P \subset T$, such that P is a 1/3-separator of T.

Proof: Let n be the number of vertices of T and r be a centroid of T. There are at most 2 branches of r, in which the number of vertices exceed $n/3$. If there is no such branch, then r is itself a 1/3-separator. Let A be a branch of r with $|V(A)| > n/3$. Since A itself is a tree with no more than $n/2$ vertices, a centroid r_a of A is a 1/2-separator of A, and each branch of r_a contains no more than $n/4$ vertices of A. If there is another branch B of r such that $|V(B)| > n/3$, a centroid r_b of B can be found such that each branch of r_b contains no more than $n/4$ vertices of B. Consider the path $P = SP_T(r_a, r) \cup SP_T(r, r_b)$. Since each branch of P contains no more than $n/3$ vertices, P is a 1/3-separator of T. Note that if B does not exist, then $SP_T(r_a, r)$ is a 1/3-separator. □

In the following paragraphs, a *path separator* of a tree T is a path and meanwhile a minimal 1/3-separator of T. Substituting $\delta = 1/3$ in Lemma 3, we obtain a lower bound of the minimum routing cost.

Corollary 5: If P is a path separator of $\widehat{T}$, then

$$C(\widehat{T}) \geq \frac{4n}{3} \sum_{v \in V} d_{\widehat{T}}(v, P) + \frac{4n^2}{9} w(P).$$

Lemma 6: There exist $r_1, r_2 \in V$ such that if $R = SP_G(r_1, r_2)$ and $T \in \text{star}(R)$, $C(T) \leq (15/8)C(\widehat{T})$.

Proof: Let P be a path separator of $\widehat{T}$ with endpoints r_1 and r_2. Since T is a general star with core R, by Lemma 2,

$$C(T) \leq 2n \sum_{v \in V(G)} d_G(v, R) + \frac{n^2}{2} w(R). \tag{10}$$

Let $S = VB(\widehat{T}, P, r_1) \cup VB(\widehat{T}, P, r_2)$ denote the set of vertices in the branches incident to the two endpoints of P. For any $v \in S$,

$$\begin{aligned} d_G(v, R) &\leq \min\{d_G(v, r_1), d_G(v, r_2)\} \\ &\leq d_{\widehat{T}}(v, P). \end{aligned}$$

For $v \notin S$,

$$\begin{aligned} d_G(v, R) &\leq \min\{d_G(v, r_1), d_G(v, r_2)\} \\ &\leq \left(d_G(v, r_1) + d_G(v, r_2)\right)/2 \\ &\leq d_{\widehat{T}}(v, P) + w(P)/2. \end{aligned}$$

Then, by Fact 1, $|S| \geq \frac{2n}{3}$, and therefore

$$\sum_{v \in V} d_G(v, R) \leq \sum_{v \in V} d_{\widehat{T}}(v, P) + (n/6) w(P). \tag{11}$$

Substituting this in (10) and recalling that $w(R) \leq w(P)$ since R is a shortest path between r_1 and r_2, we have

$$C(T) \leq 2n \sum_{v \in V} d_{\widehat{T}}(v, P) + (5n^2/6) w(P). \tag{12}$$

Comparing with the lower bound in Corollary 5, we obtain

$$C(T) \leq \max\{3/2, 15/8\} C(\widehat{T}) = (15/8) C(\widehat{T}).$$

$\square$

By Lemma 6 we can have a 15/8-approximation algorithm for the MRCT problem. For every r_1 and r_2 in V, we construct a shortest path $R = SP_G(r_1, r_2)$ and a general star $T \in \text{star}(R)$ including the degenerated cases $r_1 = r_2$. The one with the minimum routing cost must be a 15/8-approximation of the MRCT. All-pairs shortest paths can be found in $O(n^3)$ time. A direct method takes $O(n \log n + m)$ time for each pair r_1 and r_2, and therefore $O(n^3 \log n + n^2 m)$ time in total. In the next lemma, it is shown that this can be done in $O(n^3)$.

Lemma 7: Let $G = (V, E, w)$. There is an algorithm which constructs a general star $T \in \text{star}(SP_G(r_1, r_2))$ for every vertex pair r_1 and r_2 in $O(n^3)$ time.

Proof: For any $r \in V$, if a general star $T \in \text{star}(SP_G(r, v))$ for each $v \in V$ can be constructed with total time complexity $O(n^2)$, then all the stars can be constructed in $O(n^3)$ time by applying the algorithm n times for each $r \in V$. By Lemma 1, a star $T \in \text{star}(SP_G(r, v))$ can be constructed in $O(n)$ time if, for every $u \in V$, a shortest path from u to $SP_G(r, v)$ is given. Define $A(u, v) = d_G(u, SP_G(r, v))$ and $B(u, v)$ to be the vertex $k \in SP_G(r, v)$ such that $SP_G(u, k) = SP_G(u, SP_G(r, v))$. Since the all-pairs shortest paths can be constructed in $O(n^2 \log n + mn)$ time at the preprocessing stage, we need to compute $A(u, v)$, as well as $B(u, v)$, in $O(n^2)$ time for all $u, v \in V$.

First, construct a shortest-paths tree S rooted at r. Let $parent(v)$ denote the parent of v in S. It is not hard to see that

$$A(u, v) = \min\{A(parent(v), u), d_G(u, v)\}$$

for $u, v \in V - \{r\}$, and $A(r, u) = d_G(r, u)$. Therefore by a top-down traversal of S, we can compute $A(u, v)$ and $B(u, v)$ for all $u, v \in V$ in $O(n^2)$ time. □

The next theorem can be derived directly from Lemmas 6 and 7.

Theorem 8: There is a 15/8-approximation algorithm for the MRCT problem with time complexity $O(n^3)$.

1.3 A 3/2-approximation algorithm

Let P be a path separator of an optimal tree. By Lemma 2, if $X \in \text{star}(P)$, then

$$C(X) \leq 2n \sum_{v \in V} d_G(v, P) + (n^2/2)w(P).$$

Since $d_G(v, P) \leq d_{\widehat{T}}(v, P)$ for any v, it can be shown that X is a 3/2-approximation solution by Corollary 5. However, it costs exponential time to try all possible paths. In the following we show that a 3/2-approximation solution can be found if, in addition to the two endpoints of P, we know a centroid of an optimal tree.

Let $P = (p_1, p_2, ..., p_k)$ be a path separator of $\widehat{T}$, $V_i = VB(T, P, p_i)$, and $n_i = |V_i|$ for $1 \leq i \leq k$. It is easy to see that a centroid must be in $V(P)$. Let p_q be a centroid of $\widehat{T}$. Construct $R = SP_G(p_1, p_q) \cup SP_G(p_q, p_k)$. We assume that R has no cycle. Otherwise, we arbitrarily remove edges to break the cycles. Obviously $w(R) \leq w(P)$. Let $T \in \text{star}(R)$. The next lemma shows the approximation ratio.

Lemma 9: $C(T) \leq (3/2)C(\widehat{T})$.

Proof: First, for any $v \in V_1 \cup V_q \cup V_k$,

$$\begin{aligned} d_G(v, R) &\leq \min\{d_G(v, p_1), d_G(v, p_q), d_G(v, p_k)\} \\ &\leq d_{\widehat{T}}(v, P). \end{aligned}$$

For $v \in \bigcup_{1<i<q} V_i$,

$$\begin{aligned} d_G(v, R) &\leq \min\{d_G(v, p_1), d_G(v, p_q)\} \\ &\leq (d_G(v, p_1) + d_G(v, p_q))/2 \\ &\leq d_{\widehat{T}}(v, P) + d_{\widehat{T}}(p_1, p_q)/2. \end{aligned}$$

Similarly, for $v \in \bigcup_{q<i<k} V_i$,

$$d_G(v, S) \leq d_{\widehat{T}}(v, P) + d_{\widehat{T}}(p_q, p_k)/2.$$

By Fact 1 and the property of a centroid, we have $\sum_{1<i<q} n_i \leq n/6$ and $\sum_{q<i<k} n_i \leq n/6$. Thus,

$$\sum_{v \in V} d_G(v, R) \leq \sum_{v \in V} d_{\widehat{T}}(v, P) + (n/12)w(P).$$

By Lemma 2 and Corollary 5,

$$\begin{aligned} C(T) &\leq 2n \sum_{v \in V} d_G(v, R) + (n^2/2)w(R) \\ &\leq 2n \sum_{v \in V} d_{\widehat{T}}(v, P) + (2n^2/3)w(P) \\ &\leq (3/2)C(\widehat{T}). \end{aligned}$$

□

Theorem 10: There is a 3/2-approximation algorithm with time complexity $O(n^4)$ for the MRCT problem.

Proof: First, the all-pairs shortest paths can be found in $O(n^2 \log n + mn)$. For every triple (r_1, r_0, r_2) of vertices, we construct $R = SP_G(r_1, r_0) \cup SP_G(r_0, r_2)$ and $T \in \text{star}(R)$ including the degenerated cases $r_i = r_j$. By Lemma 9, at least one of these stars is a 3/2-approximation solution of the MRCT problem, and we can choose the one with the minimum routing cost. For the time complexity, we show that each star can be constructed in $O(n)$ time. By Lemma 1, a $T \in \text{star}(R)$ can be constructed in $O(n)$ time if for every $v \in V$, a shortest path from v to R is given. Define $A(i,j,k) = d_G(i, SP_G(j,k))$ and $B(i,j,k)$ to be the vertex in $SP_G(j,k)$ which is closest to i. It is easy to see that $A(i,j,k)$ and $B(i,j,k)$ can be computed in $O(n^4)$ time.[1] For any $R = SP_G(r_1, r_0) \cup SP_G(r_0, r_2)$, since

$$d_G(v,R) = \min\{A(v,r_1,r_0), A(v,r_0,r_2)\},$$

$d_G(v,R)$ as well as the vertex in R closest to v can be computed in total $O(n^4)$ time for all $v \in V$ and for all such R at a preprocessing step. Finally, for any spanning tree T, we can compute $C(T)$ in $O(n)$ time. So the total time complexity is $O(n^4)$. □

[1]Remark: It can be computed in $O(n^3)$ time by dynamic programming. However the total time complexity is still $O(n^4)$.