國立臺灣大學電機資訊學院資訊工程學系
碩士論文
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

在加權樹上基於郵政模型的廣播雙中心問題
The Broadcasting 2-Center Problem in Weighted Trees under the Postal Model

徐展鴻
Chan-Hung Hsu

指導教授: 陳健輝博士
林清池博士
Advisors: Gen-Huey Chen, Ph.D.
Ching-Chi Lin, Ph.D.

中華民國 109 年 6 月
June, 2020

doi:10.6342/NTU202001303

誌謝

感謝林清池教授，讓我能有參加會議和投稿期刊的經驗。您常與我分享許多人生體驗，也給我許多鼓勵，讓我在每週的討論除了學業上的進步，也學習到許多生命課題。感謝陳健輝教授，准許我僅用一年的時間完成我的碩士學位，讓我更彈性的規劃我的人生。在專業上，您教會我歸納重點與報告的技巧；為人處世上，您讓我學會更堅定的追求目標，更愉悅的對待生活發生的一切。

另外，我要感謝實驗室的各位同學們。感謝承孝學長提供了演算法的雛形，讓此篇研究可以順利的完成。感謝耿健時常陪我討論許多競賽題目，一起思考許多難題，讓我參加培訓班能夠順利；感謝馭銓時常關心我的生活，和我分享喜怒哀樂，面對各種難題；感謝大有，光哲，胤竹，維良，御恆，子瑋，啟元，英傑，語宸，耕竹，子平，柏勳等人，陪我一起吃飯玩耍修課想論文，度過枯燥的研究生生活。

最後我要感謝我的父母，總是讓我無憂無慮的做我想做的事情，也總是支持我所做的決定。感謝我的姊姊，陪我度過成長的不同階段，讓我的生活總是如此有趣。

doi:10.6342/NTU202001303

摘要

廣播問題是在給定的圖中找尋廣播起始點，使得圖上最長通訊時間最小化。在本篇論文中，我們考慮加權樹上基於郵政模型的廣播問題。對於單中心的廣播問題，Su, Lin, and Lee 提出了線性時間複雜度的演算法。本篇論文更進一步提出線性時間複雜度的演算法，將其結果延伸到廣播雙中心問題。

我們觀察到廣播過程中會存在一條未使用到的邊，並證明最佳解中未使用到的邊會落在一條特定的路徑上。接著利用相鄰子樹的重疊性質減少重複計算，計算該路徑上每個邊兩側子樹的廣播時間，以找出廣播雙中心的位置。

關鍵字：廣播問題、雙中心、郵政模型、合成函數。

doi:10.6342/NTU202001303

Abatract

We consider the broadcasting 2-center problem in weighted trees under the postal model in this thesis. We observe that there is always an edge not used during the broadcast process. Further, we prove that the unused edge in the optimal solution will lie on a specific path structure. By computing all the broadcast time for subtrees in the both side of each edge on the path, we propose an $O(n)$ time algorithm for solving the broadcasting 2-center problem.

Keywords: broadcasting problem、2-center、postal model、function composition.

doi:10.6342/NTU202001303

Contents

doi:10.6342/NTU202001303

List of Figures

doi:10.6342/NTU202001303

Chapter 1

Introduction

We consider the broadcasting problem under the postal model, which distinguishes the broadcast process into two parts, connection and transmission. In the postal model, the time for connection is a constant $\alpha > 0$, and the time for transmission varies according to the edge weight. A vertex starts to broadcast messages to its neighbors if it is a broadcast center or it receives a message from its neighbors. To broadcast a message to a neighbor, a vertex should take α time to set up the connection first and then take the transmission time to broadcast according to the edge weight. At any time, a vertex can only set up a connection to one of its neighbor. However, it can transmit messages to multiple neighbors simultaneously whenever the connections to those neighbors are set up (refer to Figure 1.1 for an illustrative example).

(a) (b)

Figure 1.1: Two transmission orders: (a) (a, b, c) and (b) (c, b, a) under the postal model.

Some notations are introduced below. The *neighborhood* $N_T(v)$ of a vertex v is the set of all vertices adjacent to v in T. Let deg(v) denote the number of neighborhood of v. For each edge $(u, v) \in E(T)$, let $w(u, v)$ denote the weight of (u, v). The removal of

doi:10.6342/NTU202001303

edge (u, v) will result in two subtrees. We use the notations $T_{u,v}$ and $T_{v,u}$ to denote the subtrees containing u and v, respectively. Clearly, we have $T_{u,v} = T - T_{v,u}$.

The broadcast time of u, denoted as $b(u, T)$, is the minimum time required to broadcast a message from u to all vertices in T. The 1-center broadcast time of T, denoted as $b_1(T)$, is the minimum time required to broadcast a message from any vertex $x \in V(T)$ to all the others in T, i.e., $b_1(T) = \min\{b(x, T)|x \in V(T)\}$. The broadcast time of u and v, denoted as $b(u, v, T)$, is the minimum time required to broadcast a message from u and v to all vertices in T simultaneously. The 2-center broadcast time of T, denoted as $b_2(T)$, is the minimum time required to broadcast a message from any two vertices $x, y \in V(T)$ to all the others in T, i.e., $b_2(T) = \min\{b(x, y, T)|x, y \in V(T)\}$.

Given a weighted tree $T = (V, E)$ with $n = |V|$ in which the weight $w(u, v)$ of each edge (u, v) represents the transmission time between them, the broadcasting 2-center problem under the postal model with a constant connection time $\alpha > 0$ is to determine the minimum time $b_2(T) = \min\{b(x, y, T)|x, y \in V(T)\}$ needed to broadcast from 2 broadcast centers to all vertices in the tree.

1.1 Previous Work

The problem for finding the broadcast time of an arbitrary vertex in general graph is proved to be NP complete [1]. Since it has been proved to be NP-complete for genreal graphs, many attempts have been made to design approximation algorithms [2, 3, 4, 5], consider the broadcasting problem in special classes of graphs [6, 7, 8, 9, 10], and provide some heuristic methods [11, 12, 13, 14, 15].

Specifically, many researchers considered the broadcasting problem in trees under several different models. For the telephone model, Slater *et al.* [1] proposed an $O(n)$ time algorithm for computing the broadcast time of a given unweighted tree. Koh *et al.* [16] extended their results to weighted trees by providing an $O(nlogn)$ time algorithm. On the other hand, for the k-broadcasting model, Harutyunyan *et al.* [17] gave an $O(n)$ time algorithm finding the k-broadcast time of a given unweighted tree. As for the postal model, Tsou *et al.* [18] provided an $O(n)$-time algorithm for solving the broadcast median prob-

doi:10.6342/NTU202001303

lem in weighted trees. Su *et al.* [19] proposed an $O(n)$-time algorithm for computing the broadcast time of a given weighted tree.

In this thesis, we extend Su *et al.* [19]'s results for the broadcasting 1-center problem to the broadcasting 2-center problem by providing a $O(n)$ time algorithm. We recall the following lemmas (due to Su *et al.* [19]), which provides some useful properties for detemining $b(v, T)$ given that $v \in V(T)$.

Lemma 1.1. (Lemma 1 in [19]) *Suppose that* $u_1, u_2, \ldots, u_k$ *are neighbors of a vertex* v *in a tree* T *such that* $w(v, u_i) + b(u_i, T_{u_i,v}) \geq w(v, u_{i+1}) + b(u_i, T_{u_{i+1},v})$ *for* $1 \leq i \leq k-1$. *Then,* $u_1, u_2, \ldots, u_k$ *is an optimal sequence of calls for* v *to broadcast messages to its neighbors. Consequently, we have* $b(v, T) = \max\{w(v, u_i) + b(u_i, T_{u_i,v}) + i\alpha \mid 1 \leq i \leq k\}$.

Lemma 1.2. (Lemma 3 in [19]) *For each edge* $(u, v) \in E(T)$*, if* $b(u, T_{u,v}) \leq b(v, T_{v,u})$*, then we have* $b(v, T) \leq b(u, T)$ *and* $b(u, T) = \alpha + w(u, v) + b(v, T_{v,u})$.

Lemma 1.3. [19] *Suppose that* $u_1, u_2, \ldots, u_k$ *are neighbors of a vertex* v *in a tree* T*, and the values* $b(u_1, T_{u_1,v}), b(u_2, T_{u_2,v}), \ldots, b(u_k, T_{u_k,v})$ *are given. Then, the value* $b(v, T) = \max\{w(v, u_i) + b(u_i, T_{u_i,v}) + i\alpha \mid 1 \leq i \leq k\}$ *can be determined in* $O(k)$ *time without sorting.*

Note that Lemma 1.3 can be obtained directly from the proof of Theorem 10 in [19].

1.2 Organization

The rest of this thesis is organized as follows. Chapter 2 gives a brief introduction of essential edge and candidate path, and defines the terms $\mathcal{P}$-broadcast time and $\mathcal{P}$-broadcast function. In Chapter 3, we propose a linear time algorithm and show its correctness and linear time complexity. In Chapter 4, we analysis the properties of $\mathcal{P}$-broadcast function in detail and use them to prove the correctness of Lemmas 3.11, 3.12, and 3.13. Finally, we give the concluding remarks in Chapter 5.

doi:10.6342/NTU202001303

Chapter 2

Candidate Paths

In this chapter, we intoduce the main idea how we solve the broadcasting 2-center problem by the observation of the candidate path. In Section 2.1, we define the concept of candidate path and show the relation between broadcasting 2-center problem and candidate path. Next, some definitions and notations are introduced in Section 2.2 in order to describe our linear time algorithm precisely.

The main difference in the broadcasting 2-center problem compared to the broadcasting 1-center problem is that we can choose 2 starting vertices in the tree to broadcast simultaneously, and each vertex in the tree can receive message from either one of these 2 starting vertices. Since no matter how we pick the 2 centers, there must be $n-2$ vertices not been broadcast in the beginning, and each time if a vertex receive a message, an edge in the tree is passed through. Therefore, an edge in the tree is not used at all during the broadcast process.

Given an optimal 2 centers position and broadcast scheme, there is an edge not used at all, which we call *essential edge*. Formally, an edge (x^*, y^*) is said to be an *essential edge* if $b_2(T) = \max\{b_1(T_{x^*,y^*}), b_1(T_{y^*,x^*})\}$. Naively, the value $b_1(T_{u,v})$ and $b_1(T_{v,u})$ can be determined in $O(n)$ time for an edge $(u, v) \in E(T)$ using the algorithm for broadcasting 1-center problem proposed by Su *et al.*[19]. It follows that the broadcasting 2-center problem can be solved in $O(n^2)$ time by testing essential edge from all $n-1$ edges in the tree T.

This thesis improves the above naive algorithm by observing that there is an optimal solution, in which the essential edge lies on the specefic path structure, which we call

doi:10.6342/NTU202001303

candidate path $\mathcal{P}$ (for essential edge).

2.1 Essential Edges and Candidate Paths

In this section, we give the formal definition of the candidate path $\mathcal{P}$, and prove that the candidate path $\mathcal{P}$ contains an essential edge in Theorem 1. Before that, we first introduce some properties of the broadcast center.

Lemma 2.1. *A vertex $k \in V(T)$ is a broadcast center if $b(k, T_{k,u}) \geq b(u, T_{u,k})$ for each vertex $u \in N_T(k)$.*

Proof. We prove the statement by showing that $b(k, T) \leq b(v, T)$ for each vertex $v \in V(T)$ using induction on $d(k, v)$, where $d(k, v)$ is the number of edges on the path from k to v in T. First, we consider the case when $d(k, v) = 1$. Since $(k, v) \in E(T)$ and $b(k, T_{k,v}) \geq b(v, T_{v,k})$, we have $b(k, T) \leq b(v, T)$ by Lemma 1.2.

Suppose that the statement holds when $d(k, v) = n$. We consider the case when $d(k, v) = n + 1$ below. Let k' and v' be the neighbor of k and v on the path from k to v. Since $T_{v,v'} \subseteq T_{k',k}$ and $T_{k,k'} \subseteq T_{v',v}$, we have $b(v, T_{v,v'}) \leq b(k', T_{k',k})$ and $b(k, T_{k,k'}) \leq b(v', T_{v',v})$ respectively. Further, the inequality $b(k', T_{k',k}) \leq b(k, T_{k,k'})$ holds as $k' \in N_T(k)$. Then, we have $b(v, T_{v,v'}) \leq b(k', T_{k',k}) \leq b(k, T_{k,k'}) \leq b(v', T_{v',v})$ and so by Lemma 1.2 we have $b(v', T) \leq b(v, T)$. Therefore, by the induction hypothesis, the statement holds as $b(k, T) \leq b(v', T) \leq b(v, T)$. $\square$

Lemma 2.1 gives a sufficient condition for the broadcast center. A vertex k is called a *prime broadcast center* if it satisfies the condition of Lemma 2.1. The concept of the prime broadcast center is firstly proposed by Tsou *et al.* [20]. Notice that we can always find a prime broadcast center k in a given weighted tree T such that $b(k, T_{k,u}) \geq b(u, T_{u,k})$ for each vertex $u \in N_T(k)$ by comparing between every two adjecent vertices. Therefore, every weighted tree T contains at least one *prime broadcast center* k.

doi:10.6342/NTU202001303

Let k be a prime broadcast center of T. Further, suppose that T is a ordered tree rooted by k, and the children $u_1, u_2, \ldots, u_k$ of an arbitrary vertex v in T are ordered such that $w(v, u_i) + b(u_i, T_{u_i,v}) \geq w(v, u_{i+1}) + b(u_{i+1}, T_{u_{i+1},v})$ for $1 \leq i \leq k-1$. A candidate edge set for an essential edge, called *candidate path* $\mathcal{P}$, is defined below. Let

$$\mathcal{P} = (x_s, \ldots, x_2, x_1, k, y_1, y_2 \ldots, y_t),$$

where x_1 and y_1 are the first and the second child of k, x_i is the first child of x_{i-1} for $2 \leq i \leq s$, and y_j is the first child of y_{j-1} for $2 \leq j \leq t$. For technical purposes, we assume that $x_1 = y_{-1}$, $x_0 = k = y_0$, and $x_{-1} = y_1$. Note that the candidate path $\mathcal{P}$ partition the original tree T into some subtrees. Let T_v be the subtree partioned by the candidate path $\mathcal{P}$ that contains the vertex v. Furthermore, for any subpath $(z_1, z_2, ..., z_k)$ of $\mathcal{P}$, we define $T(z_1, z_k) = T(z_k, z_1) = T_{z_1} \cup T_{z_2} \cup ... \cup T_{z_k} \cup (z_1, z_2, ..., z_k)$ to be the subtree containing $T_{z_1}, T_{z_2}, ..., T_{z_k}$ (refer to Figure 2.1).

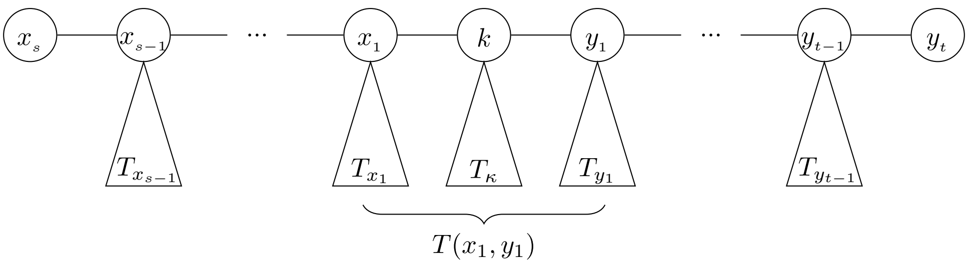

Figure 2.1: A general view of a candidate path.

Intuively, the edge with larger weight is likely to be an essential edge, on the other hand, the position of the essential edge should be close to the broadcast center k. Actually, 2 properties should be considered at the same time, and the *candidate path* $\mathcal{P}$ in some way makes a balance between them. Refer to Figure 2.2. Two examples are given assume that we have $\alpha = 1$. One can verify that the candidate path $\mathcal{P}$ are those edges with thick line, besides, the edge (b_1, b_2) is an essential edge in (a), and the edge (a_1, a_2) is an essential edge in (b). In both cases, there is an essential edge lieing on the candidate path. Before proving this property of the candidate path, we first show the following lemma.

doi:10.6342/NTU202001303

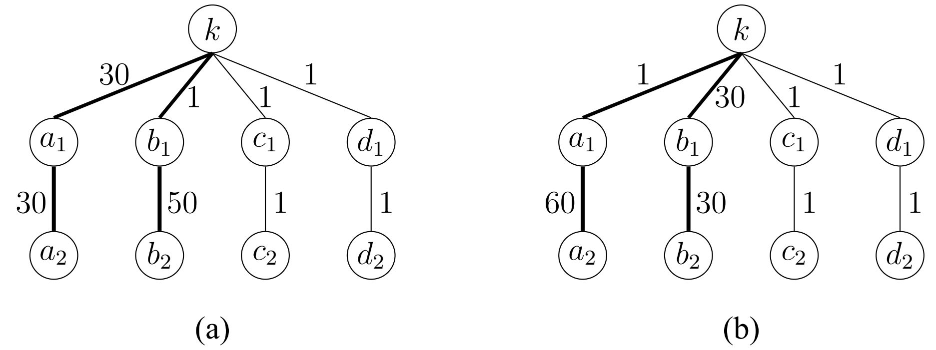

Figure 2.2: Two candidate paths containing essential edges: (a) (b_1, b_2) and (b) (a_1, a_2).

Lemma 2.2. *For an edge $(u, v) \in E(T)$, if k is a prime broadcast center of T lieing in $T_{u,v}$ and k' is a prime broadcast center of $T_{u,v}$. Then, the vertex k lies on the path from u to k'.*

Proof. If $k = u$, then the statement holds immediately. Otherwise, let $T' = T_{u,v}$ and suppose that p is the neighbor of k on the path from k to u in T'. We prove the statement by showing that $k' \in V(T'_{k,p})$. Since T' is a subtree of T, we have $b(p, T'_{p,k}) \leq b(p, T_{p,k})$.

We first consider the case when $b(p, T'_{p,k}) = b(p, T_{p,k})$. Notice that in this case, we have $b(c, T'_{c,k}) = b(c, T_{c,k})$ for each vertex c in $N_{T'}(k)$, and so $b(k, T'_{k,c}) = b(k, T_{k,c})$ holds for each vertex c in $N_{T'}(k)$ by Lemma 1.1. Thus, we have $b(k, T'_{k,c}) \geq b(c, T'_{c,k})$ for each vertex c in $N_{T'}(k)$ by the fact that $b(k, T_{k,c}) \geq b(c, T_{c,k})$ for each vertex c in $N_T(k)$. It follows that k is a prime broadcast center of T', i.e., $k' = k$. Then, the statement holds as $k \in V(T'_{k,p})$.

Next, we consider the case when $b(p, T'_{p,k}) < b(p, T_{p,k})$. Assume to the contrary that $k' \in V(T'_{p,k})$ and q is the neighbor of k' on the path from k' to k in T'. One can see that we have

$$
\begin{aligned}
b(k', T'_{k',q}) &\leq b(p, T'_{p,k}) && (T'_{k',q} \subseteq T'_{p,k}) \\
&< b(p, T_{p,k}) \\
&\leq b(k, T_{k,p}) && \text{(by Lemma 2.1)} \\
&= b(k, T'_{k,p}) && (T_{k,p} = T'_{k,p}) \\
&\leq b(q, T'_{q,k'}), && (T'_{k,p} \subseteq T'_{q,k'})
\end{aligned}
$$

doi:10.6342/NTU202001303

contradicting to the fact that k' is a prime broadcast center of T'. Hence, $k' \in V(T'_{k,p})$. $\square$

Next, we prove that the candidate path $\mathcal{P}$ indeed contains an essential edge. Besides, we further prove that the exact position of the candidate path $\mathcal{P}$ can be determined in $O(n)$ time. Recall that an edge (x^*, y^*) is said to be an *essential edge* if $b_2(T) = \max\{b_1(T_{x^*,y^*}), b_1(T_{y^*,x^*})\}$. In the following proof, we use the fact that if $b(T_{x,y}) \leq b(T_{u,v})$ and $b(T_{y,x}) \leq b(T_{u,v})$, then (u, v) is an essential edge implies (x, y) is also an essential edge.

Theorem 1. *The candidate path $\mathcal{P}$ of a tree T contains an essential edge (x^*, y^*).*

Proof. If (x^*, y^*) is in $\mathcal{P}$, then we are done. Otherwise, without loss of generousity, we assume that there is a prime broadcast center $k \in V(T_{x^*,y^*})$ and the edge (x^*, y^*) is contained in the subtree T_{y_j} for some $j > 0$. We prove the statement by showing that the edge (y_j, y_{j+1}) is also an essential edge of T.

We first consider the case $x^* = y_j$. Note that $b(T_{y_{j+1},y_j}) \leq b(T_{y_j,y^*})$ since T_{y_{j+1},y_j} is a subtree of T_{y_j,y^*}. Suppose that k' is a prime broadcast center of T_{y_j,y^*}, then by Lemma 2.2, we have $k' \in T_{y_0,y_1}$. By the definition of y_{j+1}, we have $b(y^*, T_{y^*,y_j}) + w(y_j, y^*) \leq b(y_{j+1}, T_{y_{j+1},y_j}) + w(y_j, y_{j+1})$, implying that $b(k', T_{y_j,y_{j+1}}) \leq b(k', T_{y_j,y^*})$ by Lemma 1.1. It follows that $b(T_{y_j,y_{j+1}}) \leq b(k', T_{y_j,y_{j+1}}) \leq b(k', T_{y_j,y^*}) = b(T_{y_j,y^*})$. Therefore, (y_j, y_{j+1}) is also an essential edge.

Next, we consider the case $x^* \neq y_j$. Suppose that p is the neighbor of y_j on the path from x^* to y_j in T_{y_j}. Note that $b(T_{y_j,p}) \leq b(T_{x^*,y^*})$ since $T_{y_j,p} \subseteq T_{x^*,y^*}$. And also, we have $b(T_{p,y_j}) \leq b(T_{y_j,y_{j+1}})$ as $T_{p,y_j} \subseteq T_{y_j,y_{j+1}}$. Besides, we have shown that $b(T_{y_j,y_{j+1}}) \leq b(T_{y_j,p})$ in the previous case. Therefore, we have $b(T_{p,y_j}) \leq b(T_{y_j,y_{j+1}}) \leq b(T_{y_j,p}) \leq b(T_{x^*,y^*})$, implying that (y_j, p) is also an essential edge.

On the other hand, if the edge (x^*, y^*) is contained in the subtree T_{y_0}, using the similar argument, one can veify that either (y_0, y_1) or (y_0, x_1) is an essential edge of T depending on the position of a prime broadcast center k' of T_{y_j,y^*}. $\square$

doi:10.6342/NTU202001303

Lemma 2.3. *The candidate path $\mathcal{P}$ can be determined in $O(n)$ time. Furthermore, for any edge $(x, y) \in E(T)$ with x lieing on $\mathcal{P}$ and y not lieing on $\mathcal{P}$, the broadcast time $b(y, T_{y,x})$ can be obtained during the process.*

Proof. We run the Algorithm BROADCAST proposed by Su *et al.* [19] once with a little modification. Note that the only vertex κ left after the while loop is indeed a prime broadcast center k of the tree T. Also, for any vertex $v \in V(T)$, we have $t(v) = b(v, T(v, v'))$, where v' is the neighbor of v such that v' is on the path from v to k. By the definition of the candidate path $\mathcal{P}$, we have k lies on $\mathcal{P}$, implying that all the broadcast time $b(y, T_{y,x})$ are obtained during the process.

To identify the exact position of the candidate path $\mathcal{P}$, we assume that T is a rooted tree with k as the root and the vertices $u_1, u_2, \ldots, u_k$ are the children of an arbitrary vertex v. During each while loop iteration in the algorithm, we record the children of v with the largest and the second largest value $w(v, u_i) + b(u_i, T_{u_i,v})$ by comparing between all children in $O(deg(v))$ time. Therefore, all the vertices in the candidate path $\mathcal{P}$ can be found out in $O(n)$ time. $\square$

2.2 More Notations

In this section, we define 3 terms, *$\mathcal{P}$-subtree*, *$\mathcal{P}$-broadcast time*, and *$\mathcal{P}$-broadcast function*. As saying previously, given a weighted tree T, we can unique determine the candidate path $\mathcal{P}$ for the essential edge.

The subtree is called a $\mathcal{P}$-subtree if it corresponds to any one of $T_{x_i,x_{i-1}}$, T_{x_{i-1},x_i}, $T_{y_j,y_{j-1}}$ or T_{y_{j-1},y_j} for some i or j. Below, we assume that two vertices u, v both lie on $\mathcal{P}$. The $\mathcal{P}$-broadcast time $b(u, v)$ is the minumum time needed to broadcast from the vertex u to all vertices in $T(u, v)$. On the other hand, the $\mathcal{P}$-broadcast function $\tilde{b}(u, v : t)$ is a function that returns the minumum time needed to broadcast from the vertex u to all vertices in $T(u, v) \cup (v, v') \cup T'$, assuming that $w(v, v') = 0$, $v' \in V(T')$, and $b(v', T') = t$. Note that the value $\tilde{b}(u, v : t)$ corresponds to some $b(u, v')$ with the same starting point u if we choose t properly.

doi:10.6342/NTU202001303

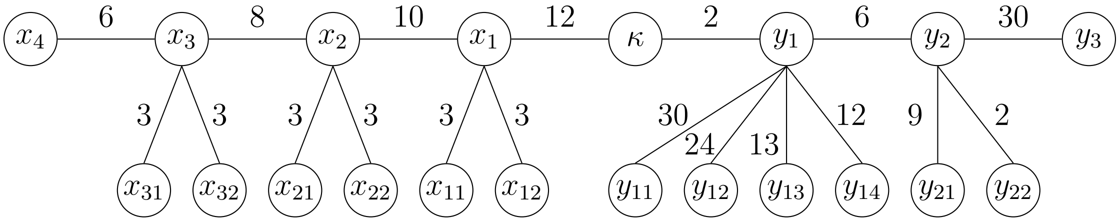

Figure 2.3: An illustrative example.

Refer to Figure 2.3 for an illustrative example. Suppose that we have $\alpha = 5$. According to Lemma 1.1, if y_2 is going to broadcast message to $T(y_2, y_3)$, it should broadcast to y_3 first, and then y_{21} and y_{22}. Therefore we have $b(y_2, y_3) = 35$ and $\tilde{b}(y_2, y_2 : 30) = 35$, that is, $\tilde{b}(y_2, y_2 : t) = b(y_2, y_3)$ if we choose $t = 30$. Furthurmore, by knowing $b(y_2, y_3) = 35$ and use Lemma 1.1 again to further calculate $b(y_1, y_3)$, one can verify that $b(y_1, y_3) = 46$ and $\tilde{b}(y_1, y_1 : 41) = 46$. We also have that $\tilde{b}(y_1, y_1 : t) = b(y_1, y_3)$ if we choose $t = 41 = 6 + 35$.

Note that the 1-center broadcast time $b_1(T_{s,t})$ doesn't specify the position of the starting vertex, which is different from the $\mathcal{P}$-broadcast time $b(u, v)$. In our linear-time algoritm introduced in the next section, we will calculate all the 1-center broadcast time $b_1(T_{s,t})$ of each $\mathcal{P}$-subtree $T_{s,t}$ with the help of $\mathcal{P}$-broadcast time $b(u, v)$ and $\mathcal{P}$-broadcast function $\tilde{b}(u, v : t)$.

doi:10.6342/NTU202001303

Chapter 3

A Linear-Time Algorithm

In this chapter, a linear time algorithm is proposed. Section 3.1 gives an overview of the linear-time algorithm. Next, the correctness and the time complexity are analysised in Section 3.2.

3.1 An Algorithm Overview

To solve the broadcasting 2-center problem, we calculate all the 1-center broadcast time $b_1(T_{s,t})$ of each $\mathcal{P}$-subtree $T_{s,t}$ in order to find out the essential edge. In algorithm 1, we use 4 for-loops to calculate those 1-center broadcast time, which corresponds to the $\mathcal{P}$-subtree T_{y_{j-1},y_j}, $T_{y_j,y_{j-1}}$, T_{x_{i-1},x_i} and $T_{x_i,x_{i-1}}$ respectively. Note that the direction of each for-loop is different. After computing all the $\mathcal{P}$-subtree broadcast time, we choose an essential edge among all edges on $\mathcal{P}$, and calculate the 2-center broadcast time $b_2(T)$.

doi:10.6342/NTU202001303

Algorithm 1 Solving the broadcasting 2-center problem.

Input: A weighted tree $T = (V, E)$.
Output: Essential edge (x^*, y^*) and 2-center broadcast time $b_2(T)$.

determine the candidate path $\mathcal{P} = (x_s, ..., x_2, x_1, k, y_1, y_2, ..., y_t)$;
for $j = t$ **downto** 1 **do**
  calculate the 1-center broadcast time $b_1(T_{y_{j-1},y_j})$;
end for
for $j = 1$ **to** t **do**
  calculate the 1-center broadcast time $b_1(T_{y_j,y_{j-1}})$;
end for
for $i = s$ **downto** 1 **do**
  calculate the 1-center broadcast time $b_1(T_{x_{i-1},x_i})$;
end for
for $i = 1$ **to** s **do**
  calculate the 1-center broadcast time $b_1(T_{x_i,x_{i-1}})$;
end for
choose an essential edge (x^*, y^*) on $\mathcal{P}$ s.t. $\max\{b_1(T_{x^*,y^*}), b_1(T_{y^*,x^*})\}$ is minimized;

return (x^*, y^*) and $b_2(T) = \max\{b_1(T_{x^*,y^*}), b_1(T_{y^*,x^*})\}$

The implementation of Algorithm 1 will be shown in Procedures 2 and 3 later. Below, we prove some useful lemmas to explain how the implementation works. First, Lemmas 3.1-3.4 show how to find the broadcast center position of each $\mathcal{P}$-subtree T_{y_{j-1},y_j}, $T_{y_j,y_{j-1}}$, T_{x_{i-1},x_i} and $T_{x_i,x_{i-1}}$.

Lemma 3.1. *Let x_k be the vertex on $\mathcal{P}$ with $0 \le k \le s-1$ satisfying $b(x_k, x_s) \ge b(x_{k-1}, y_{j-1})$ and $b(x_k, y_{j-1}) \ge b(x_{k+1}, x_s)$. Then, the vertex x_k is a prime boradcast center of the $\mathcal{P}$-subtree T_{y_{j-1},y_j}.* (The condition $b(x_k, x_s) \ge b(x_{k-1}, y_{j-1})$ is no need if $k = 0$ and $j = 1$.)

Proof. Suppose that $T' = T_{y_{j-1},y_j}$. By Lemma 2.1, it suffices to show that the vertex x_k satisfies $b(x_k, T'_{x_k,u}) \ge b(u, T'_{u,x_k})$ for all vertex $u \in N_{T'}(x_k)$. For the case that $u = x_{k+1}$, we have $b(x_k, T'_{x_k,u}) = b(x_k, T'_{x_k,x_{k+1}}) = b(x_k, y_{j-1}) \ge b(x_{k+1}, x_s) = b(x_{k+1}, T'_{x_{k+1},x_k}) = b(u, T'_{u,x_k})$. Similarly, if $u = x_{k-1}$, we have $b(x_k, T'_{x_k,u}) = b(x_k, T'_{x_k,x_{k-1}}) = b(x_k, x_s) \ge b(x_{k-1}, y_{j-1}) = b(x_{k-1}, T'_{x_{k-1},x_k}) = b(u, T'_{u,x_k})$.

Finally, we consider the case that $u \in N_{T'}(x_k) \cap V(T_{x_k})$, i.e., u is not on the candidate

doi:10.6342/NTU202001303

path. In this case, we have

$$\begin{aligned}
b(x_k, T'_{x_k,u}) &\geq \alpha + w(x_k, x_{k+1}) + b(x_{k+1}, T'_{x_{k+1},x_k}) && \text{(broadcast to subtree } T'_{x_{k+1},x_k}\text{)} \\
&= \alpha + w(x_k, x_{k+1}) + b(x_{k+1}, T_{x_{k+1},x_k}) && (T'_{x_{k+1},x_k} = T_{x_{k+1},x_k}) \\
&\geq \alpha + w(x_k, u) + b(u, T_{u,x_k}) && \text{(definition of } x_{k+1}\text{)} \\
&> b(u, T_{u,x_k}) \\
&= b(u, T'_{u,x_k}), && (T'_{u,x_k} = T_{u,x_k})
\end{aligned}$$

which completes the proof. □

Lemma 3.2. *Let y_k be the vertex on $\mathcal{P}$ with $j \leq k \leq t-1$ satisfying $b(y_k, y_t) \geq b(y_{k-1}, y_j)$ and $b(y_k, y_j) \geq b(y_{k+1}, y_t)$. Then, the vertex y_k is a prime boradcast center of the $\mathcal{P}$-subtree $T_{y_j,y_{j-1}}$.* (The condition $b(y_k, y_t) \geq b(y_{k-1}, y_j)$ is no need if $k = j$.)

Lemma 3.3. *Let y_k be the vertex on $\mathcal{P}$ with $0 \leq k \leq t-1$ satisfying $b(y_k, y_t) \geq b(y_{k-1}, x_{i-1})$ and $b(y_k, x_{i-1}) \geq b(y_{k+1}, y_t)$. Then, the vertex y_k is a prime boradcast center of the $\mathcal{P}$-subtree T_{x_{i-1},x_i}.* (The condition $b(y_k, y_t) \geq b(y_{k-1}, x_{i-1})$ is no need if $k = 0$ and $i = 1$.)

Lemma 3.4. *Let x_k be the vertex on $\mathcal{P}$ with $i \leq k \leq s-1$ satisfying $b(x_k, x_s) \geq b(x_{k-1}, x_i)$ and $b(x_k, x_i) \geq b(x_{k+1}, x_s)$. Then, the vertex x_k is a prime boradcast center of the $\mathcal{P}$-subtree $T_{x_i,x_{i-1}}$.* (The condition $b(x_k, x_s) \geq b(x_{k-1}, x_i)$ is no need if $k = i$.)

Using the similar arguments stated in Lemma 3.1, one can also prove Lemmas 3.2-3.4, hence, we omit the proofs. The implementation of the algorithm utilizes Lemmas 3.1-3.4 to find a prime broadcast center of each $\mathcal{P}$-subtree. Next, by utilizing the position of a prime broadcast center, Lemmas 3.5-3.8 are introduced to calculate the broadcast time of each $\mathcal{P}$-subtree.

Lemma 3.5. *Let x_k be a prime broadcast center of the $\mathcal{P}$-subtree T_{y_{j-1},y_j}. Then, $b_1(T_{y_{j-1},y_j}) = \min\{u, v\}$, where $u = \max\{\alpha + w(x_k, x_{k-1}) + b(x_{k-1}, y_{j-1}),\ \alpha + b(x_k, x_s)\}$ and $v = \max\{\alpha + w(x_k, x_{k+1}) + b(x_{k+1}, x_s),\ \alpha + b(x_k, y_{j-1})\}$.* (The value $u = \infty$ if $k = 0$ and $j = 1$.)

doi:10.6342/NTU202001303

Proof. By definition of the candidate path $\mathcal{P}$, we have $w(x_k, x_{k+1}) + b(x_{k+1}, T_{x_{k+1},x_k}) \geq w(x_k, u) + b(u, T_{u,x_k})$ for all $u \in N_T(x_k) \cap V(T_{x_k})$. Suppose that $T' = T_{y_j,y_{j+1}}$. Since $T_{x_{k+1},x_k} = T'_{x_{k+1},x_k}$ and $T_{u,x_k} = T'_{u,x_k}$, the optimal seqence of call for x_k to broadcast message to its neighbor in T' must start with either x_{k+1} or x_{k-1} by Lemma 1.1.

If it starts with x_{k+1}, then x_k takes α time to set up connection to x_{k+1}, and x_{k+1} will receive message just at time $\alpha + w(x_k, x_{k+1})$. Therefore, the time needed to broadcast to all vertices will be max$\{\alpha + w(x_k, x_{k+1}) + b(x_{k+1}, x_s),\ \alpha + b(x_k, y_{j-1})\}$. Similarly, if it starts with x_{k-1}, then x_k takes α time to set up connection to x_{k-1}, and x_{k-1} will receive message just at time $\alpha + w(x_k, x_{k-1})$. Therefore, the time needed to broadcast to all vertices will be max$\{\alpha + w(x_k, x_{k-1}) + b(x_{k-1}, y_{j-1}),\ \alpha + b(x_k, x_s)\}$. □

Lemma 3.6. *Let y_k be a prime broadcast center of the $\mathcal{P}$-subtree $T_{y_j,y_{j-1}}$. Then, $b_1(T_{y_j,y_{j-1}}) =$ min$\{u, v\}$, where $u =$ max$\{\alpha + w(y_k, y_{k-1}) + b(y_{k-1}, y_j),\ \alpha + b(y_k, y_t)\}$ and $v =$ max$\{\alpha + w(y_k, y_{k+1}) + b(y_{k+1}, y_t),\ \alpha + b(y_k, y_j)\}$.* (The value $u = \infty$ if $k = j$.)

Lemma 3.7. *Let y_k be a prime broadcast center of the $\mathcal{P}$-subtree T_{x_{i-1},x_i}. Then, $b_1(T_{x_{i-1},x_i}) =$ min$\{u, v\}$, where $u =$ max$\{\alpha + w(y_k, y_{k-1}) + b(y_{k-1}, x_{i-1}),\ \alpha + b(y_k, y_t)\}$ and $v =$ max$\{\alpha + w(y_k, y_{k+1}) + b(y_{k+1}, y_t),\ \alpha + b(y_k, x_{i-1})\}$.* (The value $u = \infty$ if $k = 0$ and $i = 1$.)

Lemma 3.8. *Let x_k be a prime broadcast center of the $\mathcal{P}$-subtree $T_{x_i,x_{i-1}}$. Then, $b_1(T_{x_i,x_{i-1}}) =$ min$\{u, v\}$, where $u =$ max$\{\alpha + w(x_k, x_{k-1}) + b(x_{k-1}, x_i),\ \alpha + b(x_k, x_s)\}$ and $v =$ max$\{\alpha + w(x_k, x_{k+1}) + b(x_{k+1}, x_s),\ \alpha + b(x_k, x_i)\}$.* (The value $u = \infty$ if $k = i$.)

Lemma 3.5 shows the broadcast time $b_1(T_{y_{j-1},y_j})$ can be determined in $O(1)$ time by knowing the value $b(x_k, x_s)$, $b(x_{k+1}, x_s)$, $b(x_k, y_{j-1})$, and $b(x_{k-1}, y_{j-1})$. Likewise, Lemmas 3.6-3.8 can be similarly proved, so we omit the detailed proofs. Below, we make use of Lemmas 3.1 and 3.5 to design Procedure 2 which implements the first for-loop in Algorithm 1. The details about how to determine the $\mathcal{P}$-broadcast time $b(x_k, x_s)$, $b(x_{k+1}, x_s)$, $b(x_k, y_{j-1})$, $b(x_{k-1}, y_{j-1})$ and the value using in the while-loop will be discussed in the next section.

doi:10.6342/NTU202001303

Procedure 2 Implementation of steps(2) - (4) in Algorithm 1

Input: A weighted tree T and the candidate path $\mathcal{P} = (x_s, ..., x_2, x_1, k, y_1, y_2, ..., y_t)$.
Output: All the $\mathcal{P}$-subtree broadcast time $b_1(T_{y_{j-1},y_j})$.

let $k \leftarrow 0$;
for $j = t$ **downto** 1 **do**
  while $b(x_k, y_{j-1}) < b(x_{k+1}, x_s)$ **do**
    let $k \leftarrow k + 1$;
  end while
  // x_k is a prime broadcast center of $\mathcal{P}$-subtree T_{y_{j-1},y_j}
  determine $b_1(T_{y_{j-1},y_j})$ using $b(x_k, x_s)$, $b(x_{k+1}, x_s)$, $b(x_k, y_{j-1})$, and $b(x_{k-1}, y_{j-1})$
end for

return $b_1(T_{y_{t-1},y_t})$, $b_1(T_{y_{t-2},y_{t-1}})$, ..., and $b_1(T_{y_0,y_1})$

In each specific iteration j in the for-loop of Procedure 2, we deal with the $\mathcal{P}$-subtree T_{y_{j-1},y_j}. First, the while-loop is executed to examine the condition of Lemma 3.1 and to find a prime broadcast center x_k of $\mathcal{P}$-subtree T_{y_{j-1},y_j}. Next, by using Lemma 3.5, the $\mathcal{P}$-subtree broadcast time $b_1(T_{y_{j-1},y_j})$ can be determined by $b(x_k, x_s)$, $b(x_{k+1}, x_s)$, $b(x_k, y_{j-1})$, and $b(x_{k-1}, y_{j-1})$.

Procedure 3 Implementation of steps(5) - (7) in Algorithm 1

Input: A weighted tree T and the candidate path $\mathcal{P} = (x_s, ..., x_2, x_1, k, y_1, y_2, ..., y_t)$.
Output: All the $\mathcal{P}$-subtree broadcast time $b_1(T_{y_j,y_{j-1}})$.

1: let $k \leftarrow 1$;
2: **for** $j = 1$ **to** t **do**
3:   **if** $k < j$, **then** let $k \leftarrow j$;   // y_k must lie on $(y_j, y_{j+1}, ..., y_t)$
4:   **while** $b(y_k, y_j) < b(y_{k+1}, y_t)$ **do**
5:     let $k \leftarrow k + 1$;
6:   **end while**
7:   // y_k is a prime broadcast center of $\mathcal{P}$-subtree $T_{y_j,y_{j-1}}$
8:   determine $b_1(T_{y_j,y_{j-1}})$ using $b(y_k, y_t)$, $b(y_{k+1}, y_t)$, $b(y_k, y_j)$, and $b(y_{k-1}, y_j)$
9: **end for**
10:
11: **return** $b_1(T_{y_1,y_0})$, $b_1(T_{y_2,y_1})$, ..., and $b_1(T_{y_t,y_{t-1}})$

In the same way, Procedure 3 is designed to implement the second for-loop in Algorithm 1. According to Lemmas 3.2 and 3.6, we determine $b_1(T_{y_j,y_{j-1}})$ one by one from $j = 1$ to $j = t$ by finding out the broadcast center in each subtree $T_{y_j,y_{j-1}}$ and computing the corresponding value $b(y_k, y_t)$, $b(y_{k+1}, y_t)$, $b(y_k, y_j)$, and $b(y_{k-1}, y_j)$. Note that the Step (3) of Procedure 3 is added since the prime broadcast center y_k must lie on $(y_j, y_{j+1}, ..., y_t)$.

Due to the symmetry of the candidate path $\mathcal{P}$, the other 2 for-loops in Algorithm 1

doi:10.6342/NTU202001303

can be implemented in the same way, therefore, we omit the details. Intuively, there may be totally $O(n^2)$ different $\mathcal{P}$-broadcast time $b(u, v)$ that needs to be determined since the maximal length of $\mathcal{P}$ can be $O(n)$. However, only $O(n)$ $\mathcal{P}$-broadcast time will be covered, and Procedures 2 and 3 can both be implemented in $O(n)$ time. We analysis the correctness and the time complexity in the next section.

3.2 Correctness and Time Complexity

In this section, we prove the correctness of Procedures 2 and 3, and show that all the $\mathcal{P}$-broadcast time $b(u, v)$ needed in the precedures can be determined in $O(n)$ time.

Lemma 3.9. *Procedure 2 returns the correct $\mathcal{P}$-subtree broadcast time $b_1(T_{y_{j-1},y_j})$.*

Proof. According to Lemmas 3.1 and 3.5, to prove the correctness of Procedure 2, it suffices to show that if the while-loop terminates during the for-loop iteration j, then the vertex x_k satisfies the condition $b(x_k, x_s) \geq b(x_{k-1}, y_{j-1})$ and $b(x_k, y_{j-1}) \geq b(x_{k+1}, x_s)$. The latter inequality $b(x_k, y_{j-1}) \geq b(x_{k+1}, x_s)$ holds due to the termination of the while-loop.

For the former inequality $b(x_k, x_s) \geq b(x_{k-1}, y_{j-1})$, we consider 2 cases depending on whether the Step (4) in the while-loop has ever been executed during the same for-loop iteration j. If the Step (4) has ever been executed, then the previous execution of the while-loop implies $b(x_{k-1}, y_{j-1}) < b(x_k, x_s)$, therefore, the former inequality holds. On the other hand, if the Step (4) has never been executed, then for the case $j = t$, we have $b(x_k, x_s) \geq b(x_{k-1}, y_j)$ due to $x_k = k$ is a prime broadcast center of T, otherwise, since the vertex x_k is also a prime broadcast center of $T_{y_j,y_{j+1}}$, we have $b(x_k, x_s) \geq b(x_{k-1}, y_j)$. It follows that $b(x_k, x_s) \geq b(x_{k-1}, y_{j-1})$ since $T(x_{k-1}, y_{j-1})$ is a subtree of $T(x_{k-1}, y_j)$, therefore, the former inequality holds. □

Lemma 3.10. *Procedure 3 returns the correct $\mathcal{P}$-subtree broadcast time $b_1(T_{y_j,y_{j-1}})$.*

Proof. According to Lemmas 3.2 and 3.6, to prove the correctness of Procedure 3, it suffices to show that if the while-loop terminates during the for-loop iteration j, then the

doi:10.6342/NTU202001303

vertex y_k satisfies the condition $b(y_k, y_t) \geq b(y_{k-1}, y_j)$ and $b(y_k, y_j) \geq b(y_{k+1}, y_t)$. The latter inequality $b(y_k, y_j) \geq b(y_{k+1}, y_t)$ holds due to the termination of the while-loop.

For the former inequality, we consider 2 cases depending on whether the Step (5) in the while-loop has ever been executed during the same for-loop iteration j. If the Step (5) has ever been executed, then the previous execution of the while-loop implies $b(y_{k-1}, y_j) < b(y_k, y_t)$, therefore, the former inequality holds. On the other hand, if the Step (5) has never been executed, then for the case $j = k$, we don't need to examine the former inequlity by Lemma 3.2, otherwise, since the vertex y_k is also a prime broadcast center of $T_{y_{j-1}, y_{j-2}}$, therefore, we have $b(y_k, y_t) \geq b(x_{k-1}, y_{j-1})$. It follows that $b(y_k, y_t) \geq b(y_{k-1}, y_j)$ since $T(y_{k-1}, y_j)$ is a subtree of $T(y_{k-1}, y_{j-1})$ and thus the former inequality holds. $\square$

Below, we introduce 3 useful lemmas showing good time property about the running time of determining the broadcast time sequences. These lemmas help a lot to prove the $O(n)$ time complexity of Procedures 2 and 3, the concept is easy to understand, but the correctness proof is technical. So, we leave detailed proofs to Section 4.3.

Lemma 3.11. *Let $(z_1, z_2, \ldots, z_m)$ be a subpath of the candidate path $\mathcal{P}$. Then, the broadcast time sequence $b(z_1, z_1), b(z_2, z_1), \ldots, b(z_m, z_1)$ can be determined in the total of $O(\sum_{i=1}^{m} deg(z_i))$ time.*

Lemma 3.12. *Let $(z_1, z_2, \ldots, z_m)$ be a subpath of the candidate path $\mathcal{P}$. Then, the broadcast time sequence $b(z_1, z_1), b(z_1, z_2), \ldots, b(z_1, z_m)$ can be determined in the total of $O(\sum_{i=1}^{m} deg(z_i))$ time.*

Lemma 3.13. *Let (z_1, z_2, ..., z_m) be a subpath of the candidate path $\mathcal{P}$. Then, the broadcast time sequence $\tilde{b}(z_{a_1}, z_1 : v_1)$, $\tilde{b}(z_{a_2}, z_1 : v_2)$, ...,$\tilde{b}(z_{a_n}, z_1 : v_n)$ can be determined in the total of $O(\sum_{i=1}^{m} deg(z_i) + n)$ time under the condition that:*

(1) *a_1, a_2, ..., a_n is an (nonstrictly) increasing sequence with $a_i \in \{1, 2, ...m\}$.*

(2) *v_1, v_2, ..., v_n is a (nonstrictly) decreasing sequence of input values.*

We now prove the $O(n)$ time complexity of Procedures 2 and 3. Roughly, we partion the running time of the procedures into two parts, determining the prime broadcast center

doi:10.6342/NTU202001303

(the while-loop) and determining the 1-center broadcast time (Step (7) in Procedure 2 and Step (8) in Procedure 3). We will show that each part takes $O(n)$ time respectively.

Lemma 3.14. *Procedure 2 can be implemented in $O(n)$ time.*

Proof. Suppose that $x_{k_t}, x_{k_{t-1}}, \ldots, , x_{k_1}$ are the corresponding prime broadcast centers of $T_{y_{t-1},y_t}, T_{y_{t-2},y_{t-1}}, \ldots, T_{y_0,y_1}$. We first consider the overall running time of the while-loop. Note that the $\mathcal{P}$-broadcast time equals to some $\mathcal{P}$-broadcast function with proper input t, i.e., $b(x_{k_j}, y_{j-1}) = \tilde{b}(x_{k_j}, k : w(k, y_1) + b(y_1, y_{j-1}))$. Since $w(k, y_1) + b(y_1, y_{t-1})$, $w(k, y_1) + b(y_1, y_{t-2})$, ..., $w(k, y_1) + b(y_1, y_1)$ is a (nonstrictly) decreasing sequnce of values and can be determined in $O(\sum_{i=1}^{t} deg(y_i)) = O(n)$ time according to Lemma 3.12, therefore, Lemma 3.13 implies that all the values $b(x_{k_j}, y_{j-1})$ needed in the while-loop can be determined in $O(\sum_{i=0}^{s} deg(x_i) + n) = O(n)$ time. Note that for the case $j = 1$, $b(x_{k_j}, y_{j-1}) = b(x_{k_1}, y_0)$ should be calculated using another $O(\sum_{i=0}^{k_1} deg(x_i)) = O(n)$ time by Lemma 3.11. On the other hand, according to Lemma 3.11, all the values $b(x_{k_j+1}, x_s)$ using in the while-loop can be determined in $O(\sum_{i=0}^{s} deg(x_i)) = O(n)$ time.

Next, we consider the running time of the Step (7). According to Lemma 3.11, all the values $b(x_{k_j}, x_s)$ and $b(x_{k_j+1}, x_s)$ needed in the Step (7) can be determined in $O(\sum_{i=0}^{s} deg(x_i)) = O(n)$ time. Furthermore, by $b(x_{k_j}, y_{j-1}) = \tilde{b}(x_{k_j}, k : w(k, y_1) + b(y_1, y_{j-1}))$ and Time Properties 2 and 3, all the values $b(x_{k_j}, y_{j-1})$ needed in the Step (7) can be determined in $O(\sum_{i=0}^{s} deg(x_i) + n) = O(n)$ time. Similarly, all the values $b(x_{k_j-1}, y_{j-1})$ needed in the Step (7) can be determined in $O(\sum_{i=0}^{s} deg(x_i) + n) = O(n)$ time as well.

Since all the other steps can be easily implemented in $O(1)$ time, we conclude that Procedure 2 can be implemented in $O(n)$ time. □

Lemma 3.15. *Procedure 3 can be implemented in $O(n)$ time.*

Proof. Suppose that $y_{k_1}, y_{k_2}, \ldots, , y_{k_t}$ are the corresponding prime broadcast centers of $T_{y_1,y_0}, T_{y_2,y_1}, \ldots, T_{y_t,y_{t-1}}$, and let $y_{pivot(0)} = y_1$. By repeatly finding the prime broadcast center of $T(y_{pivot(i-1)}, y_t)$, we suppose that $y_{pivot(i)}$ is the prime broadcast center of $T(y_{pivot(i-1)}, y_t)$ for $2 \leq i \leq n-1$, and $y_{pivot(n)} = y_{t-1}$ is the prime broadcast center of

doi:10.6342/NTU202001303

$T(y_{pivot(n-1)}, y_t)$ (, one can verify that it will end up at y_{t-1}). For technical purpose, we assume that $y_{pivot(n+1)} = y_{t-1}$.

We first consider the for-loop iterations $pivot(0), pivot(1), \ldots, pivot(n)$. According to Lemma 3.11, $b(y_{pivot(i)}, y_{pivot(i)})$, $b(y_{pivot(i)+1}, y_{pivot(i)})$, ..., $b(y_{pivot(i+1)}, y_{pivot(i)})$ can be determined in $O(\sum_{j=pivot(i)}^{pivot(i+1)} deg(y_j))$ time for $0 \leq i \leq n-1$. Hence, the corresponding value $b(y_{k_j}, y_j)$ using in testing the while-loop of these iterations can be determined in $O(\sum_{i=0}^{n-1} \sum_{j=pivot(i)}^{pivot(i+1)} deg(y_j)) = 2O(\sum_{j=0}^{t} deg(y_j)) = O(n)$ time. Also, one can see that all the values $b(y_{k_j}, y_j)$ and $b(y_{k_j-1}, y_j)$ needed in the Step (8) of these iterations is obtained at the same time.

Next, we consider the other for-loop iterations. For $pivot(i) < j < pivot(i+1)$, the prime broadcast center y_{k_j} of $T_{y_j, y_{j-1}}$ will lie on $(y_{pivot(i+1)}, \ldots, y_{pivot(i+2)})$, and we have $b(y_{k_j}, y_j) = \tilde{b}(y_{k_j}, y_{pivot(i+1)} : w(y_{k_1-1}, y_{k_1}) + b(y_{k_1-1}, y_j))$. Therefore, using the similar arguments in the proof of Lemma 3.14, one can prove that the values $b(y_{k_j}, y_j)$ needed in testing the while-loop and the values $b(y_{k_j}, y_j)$ and $b(y_{k_j-1}, y_j)$ needed in the Step (8) of these iterations can be determined in $O(\sum_{j=pivot(i)}^{pivot(i+2)} deg(y_j))$ time. So, the total running time in all these for-loop iterations is $O(\sum_{i=0}^{n-1} \sum_{j=pivot(i)}^{pivot(i+2)} deg(y_j)) = 2O(\sum_{j=0}^{t} deg(y_j)) = O(n)$.

Since all the other steps can be easily implemented in $O(1)$ time, we conclude that Procedure 3 can be implemented in $O(n)$ time. □

Theorem 2. *Algorithm 1 solves the broadcasting 2-center problem in $O(n)$ time.*

Proof. The correctness is directly derived from Theorem 1, Lemmas 3.9-3.10, and the symmetry of the candidate path $\mathcal{P}$. Below, we discuss the running time. The candidate path $\mathcal{P}$ can be constructed in $O(n)$ time by Lemma 2.3. According to Lemmas 3.14-3.15 and the symmetry of the candidate path $\mathcal{P}$, all the for-loops in Algorithm 1 can be implemented in $O(n)$ time. Step (14) can also be done by a simple comparison in $O(n)$ time. Therefore, Algorithm 1 runs in $O(n)$ time. □

doi:10.6342/NTU202001303

Chapter 4

$\mathcal{P}$-Broadcast Functions

Some good properties of $\mathcal{P}$-broadcast function are shown in this chapter, which help a lot in proving Lemmas 3.12 and 3.13. In Section 4.1, we establish some fundamental characteristics of $\mathcal{P}$-broadcast function. Next, we discuss how to construct and merge the records of $\mathcal{P}$-broadcast functions in Section 4.2. Finally, the correctness proofs of Lemmas 3.12 and 3.13 are provided in Section 4.3.

In this chapter, we assume that all the values $b(y, T_{y,x})$ are already determined where x is any vertex on the candidate path $\mathcal{P}$ and y is a neighbor of x in T_x. According to Lemma 2.3, these values can be determined in advance in $O(n)$ time.

doi:10.6342/NTU202001303

4.1 Properties of $\mathcal{P}$-Broadcast Functions

Recall that the $\mathcal{P}$-broadcast function $\tilde{b}(u, v : t)$ is a function that returns the minumum time needed to broadcast from the vertex u to all vertices in $T(u, v) \cup (v, v') \cup T'$, assuming that $w(v, v') = 0$, $v' \in V(T')$, and $b(v', T') = t$.

Lemma 4.1. *The broadcast function $\tilde{b}(u, v : t)$ is continuous, piecewise linear, and the slope of each piece is either 0 or 1.*

Proof. We prove the statement by induction on $d(u, v)$, where $d(u, v)$ denotes the number of edges on the path from u to v. We first consider the case that $d(u, v) = 0$, i.e., we consider $\tilde{b}(u, u : t)$. Let $u_1, u_2, ..., u_k$ be the neighbors of u in T_u and u' be the additional neighbor of u with $w(u, u') = 0$ and $b(u', T') = t$. Moreover, let $time_i = w(u, u_i) + b(u_i, T_{u_i,u})$ for $1 \leq i \leq k$ with $time_i \geq time_{i+1}$ for $1 \leq i \leq k-1$. According to Lemma 1.1, when t lies in any specific interval $[0, time_k)$, $[time_k, time_{k-1})$, ... , $[time_2, time_1)$, $[time_1, \infty)$ the optimal sequnce of calls from u to broadcast messages to its neighbors remains the same, implying that $\tilde{b}(u, u : t)$ can be determined by picking the max value of some constant value and one linear function of slope 1. So, the statement holds when t lies on these intervals, on the other hand, one can verify that $\tilde{b}(u, u : t)$ is continuous on each boundary point $time_1$, $time_2$, ..., $time_k$.

Next, suppose that the statement holds for $d(u, v) = k$, we consider the case $d(u, v) = k+1$ below. Let x be the neighbor of u on the path from u to v. By the induction hypothesis, both the broadcast function $\tilde{b}(u, u : t)$ and $\tilde{b}(x, v : t)$ are continuous, piecewise linear, and the slope of each piece is either 0 or 1. Since $\tilde{b}(u, v : t) = \tilde{b}(u, u : w(u, x) + \tilde{b}(x, v : t))$, the broadcast function $\tilde{b}(u, v : t)$ is continuous and piecewise linear. One can see that the slope of a piece in $\tilde{b}(u, v : t)$ is 1 if the slope of the corresponding pieces in $\tilde{b}(u, u : t)$ and $\tilde{b}(x, v : t)$ are both 1, otherwise, the slope of that piece is 0. Therefore, the lemma holds. $\square$

doi:10.6342/NTU202001303

Lemma 4.2. *Let* $(e, \tilde{b}(u, v : e))$ *be the endpoint on the last piece of the broadcast function* $\tilde{b}(u, v : t)$*. Then, we have* $\tilde{b}(u, v : e) - \tilde{b}(u, v : 0) \leq \alpha$*.*

Proof. We prove the statement by induction on $d(u, v)$, where $d(u, v)$ denotes the number of edges on the path from u to v. We first consider the case that $d(u, v) = 0$, i.e., we consider $\tilde{b}(u, u : t)$. Let $u_1, u_2, ..., u_k$ be the neighbors of u in T_u and u' be the additional neighbor of u with $w(u, u') = 0$ and $b(u', T') = t$. Moreover, let $time_i = w(u, u_i) + b(u_i, T_{u_i,u})$ for $1 \leq i \leq k$ with $time_i \geq time_{i+1}$ for $1 \leq i \leq k - 1$.

According to Lemma 1.1, the optimal sequnce of calls for u is $u_1, \ldots, u_k, u'$ when $t = 0$, implying that $\tilde{b}(u, v : 0) = \max\{t_1, t_2\}$ with $t_1 = \max\{time_i + i\alpha | 1 \leq i \leq k\}$ and $t_2 = 0 + (k + 1)\alpha$. On the other hand, since $(e, \tilde{b}(u, v : e))$ lies on the endpoint on the last piece, the optimal sequnce of calls for u is $v, u_1, \ldots, u_k$ when $t = e$. Besides, the slope on the left side of e is 0 and the slope on the right side of e is 1, implying that $\tilde{b}(u, v : e) = e + \alpha = \max\{time_i + (i + 1)\alpha | 1 \leq i \leq k\}$. To sum up, since $t_1 = \tilde{b}(u, v : e) - \alpha$, we have $\tilde{b}(u, v : e) - \tilde{b}(u, v : 0) \leq \alpha$ with equality if and only if $t_1 \geq t_2$.

Next, suppose that the statement holds for $d(u, v) = k$, we consider the case $d(u, v) = k + 1$ below. Let x be the neighbor of u on the path from u to v, and let $(e_1, \tilde{b}(u, u : e_1))$ and $(e_2, \tilde{b}(x, v : e_2))$ be the corresponding endpoints on the last piece of $\tilde{b}(u, u : t)$ and $\tilde{b}(x, v : t)$. By the induction hypothesis, we have $\tilde{b}(u, u : e_1) - \tilde{b}(u, u : 0) \leq \alpha$ and $\tilde{b}(x, v : e_2) - \tilde{b}(x, v : 0) \leq \alpha$. Note that $\tilde{b}(u, v : t) = \tilde{b}(u, u : w(u, x) + \tilde{b}(x, v : t))$, and the slope of a piece in $\tilde{b}(u, v : t)$ is 0 if and only if any slope of the corresponding piece in $\tilde{b}(u, u : t)$ and $\tilde{b}(x, v : t)$ is 0. Since the slope on the left side of e in $\tilde{b}(u, v : t)$ is 0, we have $e \leq e_2$ or $w(u, x) + \tilde{b}(x, v : e) \leq e_1$. If $e \leq e_2$, we have $(\tilde{b}(x, v : e) + w(u, x)) - (\tilde{b}(x, v : 0) + w(u, x)) \leq \alpha$ by the induction hypothesis on $\tilde{b}(x, v : t)$ and hence $\tilde{b}(u, v : e) - \tilde{b}(u, v : 0) \leq \alpha$. Otherwise, if $w(u, x) + \tilde{b}(x, v : e) \leq e_1$, we have $\tilde{b}(u, u : \tilde{b}(x, v : e) + w(u, x)) - \tilde{b}(u, u, \tilde{b}(x, v : 0) + w(u, x)) \leq \alpha$ by the induction hypothesis on $\tilde{b}(u, u : t)$, implying that $\tilde{b}(u, v : e) - \tilde{b}(u, v : 0) \leq \alpha$. Therefore, the lemma holds. □

doi:10.6342/NTU202001303

Lemma 4.3. *Let (l, r) be an interval of slope 0 piece between some pieces of slope 1 in the broadcast function $\tilde{b}(u, v : t)$. Then, we have $r - l \geq \alpha$.*

Proof. We prove the statement by induction on $d(u, v)$, where $d(u, v)$ denotes the number of edges on the path from u to v. We first consider the case that $d(u, v) = 0$, i.e., we consider $\tilde{b}(u, u : t)$. Let $u_1, u_2, ..., u_k$ be the neighbors of u in T_u and u' be the additional neighbor of u with $w(u, u') = 0$ and $b(u', T') = t$. Moreover, let $time_i = w(u, u_i) + b(u_i, T_{u_i,u})$ for $1 \leq i \leq k$ with $time_i \geq time_{i+1}$ for $1 \leq i \leq k - 1$. According to Lemma 1.1 and the piece of slope 0 on (l, r) lies between some pieces of slope 1, we suppose that the optimal sequnce of calls for u when $t = l$ is $u_1, ..., u_i, u', u_{i+1}, ..., u_k$ with $time_i = l > time_{i+1}$, and the optimal sequnce of calls for u when $t = r$ is $u_1, ..., u_j, u', u_{j+1}, ..., u_k$ with $time_j > r > time_{j+1}$. Clearly, as $time_j > r > l = time_i$, we have $j < i$. Besides, by Lemma 1.1, we have $\tilde{b}(u, u : l) = l + (i + 1)\alpha$ and $\tilde{b}(u, u : r) = r + (j + 1)\alpha$. Therefore, since the slope of the piece on (l, r) is 0, we have $l + (i + 1)\alpha = r + (j + 1)\alpha$, which implies that $r - l = (i - j)\alpha \geq \alpha$.

Next, suppose that the statement holds for $d(u, v) = k$, we consider the case $d(u, v) = k + 1$ below. Let x be the neighbor of u on the path from u to v. Since $\tilde{b}(u, v : t) = \tilde{b}(u, u : w(u, x) + \tilde{b}(x, v : t))$, the slope of a piece (l, r) in $\tilde{b}(u, v : t)$ is 0 if and only if the slope of the piece (l, r) in $\tilde{b}(x, v : t)$ is 0 or the slope of the piece $(w(u, x) + \tilde{b}(x, v : l), w(u, x) + \tilde{b}(x, v : r))$ in $\tilde{b}(u, u : t)$ is 0. Therefore, by the induction hypothesis, the statment for $\tilde{b}(x, v : t)$ holds, implying that the statement also holds for $\tilde{b}(u, v : t)$. ☐

Below, we give some graph examples about the of $\mathcal{P}$-broadcast function $\tilde{b}(u, v : t)$ to have a better understanding of these lemmas. Refer to Figure 4.1, where some graphs of $\mathcal{P}$-broadcast functions in Figure 2.3 are shown. One can see that each of these $\mathcal{P}$-broadcast function consists of some pieces of slope 0 or slope 1, and Lemmas 4.2 and 4.3 also hold on these examples in the same time.

Let $(e, \tilde{b}(u, v : e))$ be the endpoint on the last piece of the broadcast function $\tilde{b}(u, v : t)$. To represent $\tilde{b}(u, v : t)$ graphically, we only need to record the starting endpoint $(0, \tilde{b}(u, v : 0))$, the ending endpoint $(e, \tilde{b}(u, v : e))$, and the (doubly linked) list of these pieces $(p_1, p_2, ..., p_k)$ with each piece $p_i = (len_i, slope_i)$ represented by its relative length

doi:10.6342/NTU202001303

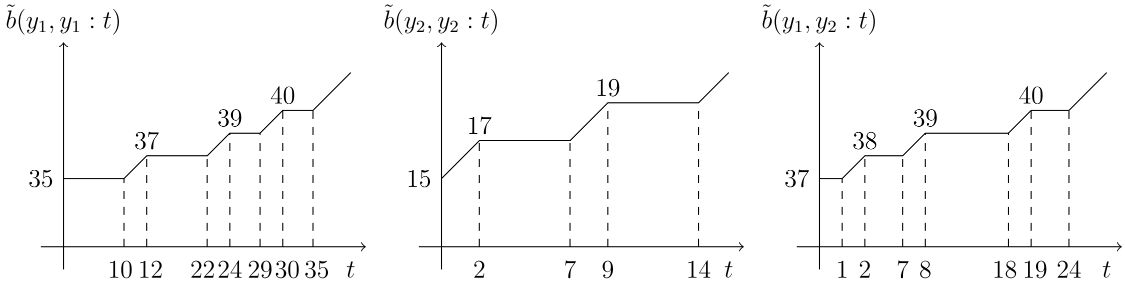

Figure 4.1: Graphs of some $\mathcal{P}$-broadcast functions in Figure 2.3.

of x coordinate and slope. Note that it doesn't need to record the last piece of slope 1 if we know the ending endpoint. In addition, we also record the $\mathcal{P}$-broadcast time $b(u, v)$.

Refer to Figure 4.2 for example, where some records associated with $\mathcal{P}$-broadcast functions in Figure 2.3 are shown. By using this kind of record, we can answer the query $\tilde{b}(u, v : x)$ of any given input x by traversing the list from the endpoint $(0, \tilde{b}(u, v : 0))$ or $(e, \tilde{b}(u, v : e))$ to the target point $(x, \tilde{b}(u, v : x))$. We make use of the record and query scheme to determine the broadcast time sequence in Lemmas 3.12 and 3.13.

$\tilde{b}(y_1, y_1 : t)$

$b(y_1, y_1)$: 35
start: $(0, 35)$
end: $(35, 40)$
list: $((10, 0), (2, 1), (10, 0), (2, 1), (5, 0), (1, 1), (5, 0))$

$\tilde{b}(y_2, y_2 : t)$

$b(y_2, y_2)$: 14
start: $(0, 15)$
end: $(14, 19)$
list: $((2, 1), (5, 0), (2, 1), (5, 0))$

$\tilde{b}(y_1, y_2 : t)$

$b(y_1, y_2)$: 37
start: $(0, 37)$
end: $(24, 40)$
list: $((1, 0), (1, 1), (5, 0), (1, 1), (10, 0), (1, 1), (5, 0))$

Figure 4.2: Records associated with some $\mathcal{P}$-broadcast functions in Figure 2.3.

4.2 Constructing the Records of $\mathcal{P}$-Broadcast Functions

In this section, 2 procedures are proposed for constructing the records of $\mathcal{P}$-broadcast functions. Procedure 4 shows how to construct the record of $\mathcal{P}$-broadcast function $\tilde{b}(u, u : t)$ given that u is a vertex on $\mathcal{P}$. Procedure 5 on the other hand shows how to construct the record of $\tilde{b}(p, s : t)$ by compositing 2 given records of $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ under the condition that $(p, ..., q, r, ..., s)$ is a subpath of $\mathcal{P}$.

doi:10.6342/NTU202001303

Given a vertex u on the candidate path $\mathcal{P}$, Precedure 4 constructs the record of $\tilde{b}(u, u : t)$ in $O(deg(u))$ time. It utilizes the non-sorting method mentioned by Su *et al.* in Theorem 10 in [19], but in which we have additional neighbor of u that needs to be considered. The correctness and the time complexity analysis of Procedure 4 are stated as the following lemmas.

Procedure 4 Constructing the record of $\tilde{b}(u, u : t)$.

Input: A weighted tree T with a vertex u on the candidate path $\mathcal{P}$.
Output: The record of the $\mathcal{P}$-broadcast function $\tilde{b}(u, u : t)$ and the value $b(u, u)$.

let $u_1, ..., u_k$ be the neighbors of u in T_u and $time_i \leftarrow w(u, u_i) + b(u_i, T_{u_i,u})$;
let $M \leftarrow \max\{time_i \mid 1 \le i \le k\}$;
create $list[j]$ for $0 \le j \le k$, and insert each vertex u_i into $list[j]$ if the condition $\alpha j \le M - time_i \le \alpha(j+1)$ holds;
let $num[j] \leftarrow |list[j]|$ and $acc[j] \leftarrow \sum_{i=0}^{j} num[i]$;
let $min[j] \leftarrow \min\{time_i \mid u_i \in list[j]\}$;
let $b(u, u) \leftarrow \max\{min[j] + \alpha acc[j] \mid 0 \le j \le k-1\}$;

let $L \leftarrow \phi$;
let $M2 \leftarrow \max\{b(u, u), (k+1)\alpha\}$;
let $point_{start} \leftarrow (0, M2)$;
if $M - (k+1)\alpha \ge 0$ **then** $L.push\_back((M - (k+1)\alpha, 0))$;

for $j = k$ **downto** 0 **do**
    let $l \leftarrow \max\{M - (j+1)\alpha, 0\}$ and $r \leftarrow M - j\alpha$;
    if $r < 0$ **then** continue;
    if $min[j] = \infty$ **then** let $min[j] \leftarrow r$;
    if $min[j] + (acc[j] + 1)\alpha \ge M2$ **then**
        let $pivot \leftarrow M2 - (acc[j] + 1)\alpha$;
        $L.push\_back((pivot - l, 0))$;
        $L.push\_back((min[j] - pivot, 1))$;
        let $M2 \leftarrow M2 + (min[j] - pivot)$;
    else
        $L.push\_back((min[j] - l, 0))$;
    end if
    $L.push\_back((r - min[j], 0))$;
end for

if $M + \alpha \le M2$ **then** $L.push\_back((M2 - \alpha - M, 0))$;
let $point_{end} \leftarrow (M2 - \alpha, M2)$;

return $(b(u, u), point_{start}, point_{end}, L)$;

doi:10.6342/NTU202001303

Lemma 4.4. *Procedure 4 constructs the record of* $\tilde{b}(u, u : t)$ *correctly.*

Proof. Let $u_1, u_2, ..., u_k$ be the neighbors of u in T_u and u' be the additional neighbor of u with $w(u, u') = 0$ and $b(u', T') = t$. The Steps (1) - (6) implement the non-sorting method mentioned by Su *et al.* in [19], in which the only difference is that we have one more list $list[k]$ since the vertex u now has $k + 1$ neighbors $u_1, ..., u_k$, and u'. It follows that the value $b(u, u)$ is determined correctly. Besides, when $t = b(u', T') = 0$, there is an optimal sequence of call for u to broadcast messages to its neighbor such that u' is the last vertex being broadcast and the broadcast order for $u_1, ..., u_k$ remains the same as the broadcast order that u only broadcasts message to $u_1, ..., u_k$. Thus, we have $\tilde{b}(u, u : 0) = \max\{b(u, u), (k + 1)\alpha\}$, and hence the coordinate of $point_{start} = (0, \tilde{b}(u, u : 0))$ is determined correctly.

Next, we prove that the list of pieces L can be constructed correctly. In each specific iteration j in the for-loop of Procedure 4, we consider the graph of $\tilde{b}(u, u : t)$ on the interval $[l_j, r_j] = [M - (j + 1)\alpha, M - j\alpha] \cap [0, \infty)$ if $[l_j, r_j] \neq \phi$. Note that $b(u, u) = \max\{min[j] + \alpha acc[j] \mid 0 \leq j \leq k - 1\}$, and we now consider $\tilde{b}(u, u : t)$ in that u has an additional vertex u' with $b(u', T') = t$. When t varies in $[l_j, r_j]$, the value of $\tilde{b}(u, u : t)$ can be determined by $\tilde{b}(u, u : t) = \max\{x, y\}$ with $x = \max\{min_t[i] + \alpha acc_t[i] \mid i \neq j\}$ and $y = min_t[j] + \alpha acc_t[j]$ where min_t and acc_t are the correponding minumum value and accumulating number of vertices under the case that $b(u', T') = t$. Clearly, x is a constant with respect to t. We now analysis the value of y when t varies. If there are some vertices in $list[j]$, then y is piecewise linear such that the slope is 1 on $[l_j, min[j]]$ and the slope is 0 on $[min[j], r_j]$. Otherwise, y is linear and the slope is 1 on $[l_j, r_j]$. It follows that the list L on each interval $[l_j, r_j]$ is determined correctly.

Using the similar arguments, one can also verify that the list of pieces L is determined correctly on the inteval $[0, M - (k + 1)\alpha]$ (if $0 \leq M - (k + 1)\alpha$) and the interval $[M, \infty)$. Therefore, the list of pieces L is constucted correctly, and the coordinate of $point_{end}$ is also determined correctly. □

doi:10.6342/NTU202001303

Lemma 4.5. *Procedure 4 constructs the record of* $\tilde{b}(u, u : t)$ *in* $O(deg(u))$ *time.*

Proof. The Steps (1) - (6) implement the non-sorting method mentioned by Su *et al.* in [19]. As shown by Su *et al.*, these stpes run in $O(deg(u))$ time. On the other hand, the for-loop in Step (13) runs in $O(k) = O(deg(u))$ time, and all the other steps can be done in $O(1)$ time. Therefore, the lemma holds. □

Procedure 5 Constructing the record of $\tilde{b}(p, s : t)$.

Input: The records of $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ with $(p, ..., q, r, ..., s)$ being a subpath of $\mathcal{P}$.
Output: The record of $\tilde{b}(p, s : t)$.

let $(e_1, \tilde{b}(p, q : e_1))$ be the ending endpoint of the broadcast function $\tilde{b}(p, q : t)$;
let $(e_2, \tilde{b}(r, s : e_2))$ be the ending endpoint of the broadcast function $\tilde{b}(r, s : t)$;
let $L \leftarrow \phi$;
let $y_1 \leftarrow \tilde{b}(r, s : 0)$ and $y_4 \leftarrow \tilde{b}(r, s : e_2)$;
let $z_1 \leftarrow y_1 + w(q, r)$, $z_4 \leftarrow y_4 + w(q, r)$, and $z_0 \leftarrow b(r, s) + w(q, r)$;

traverse $\tilde{b}(p, q : t)$ from $(0, \tilde{b}(p, q : 0))$ to the right to $(z_4, \tilde{b}(p, q : z_4))$;
let $b(p, s) \leftarrow (0, \tilde{b}(p, q : z_0))$;
let $point_{start} \leftarrow (0, \tilde{b}(p, q : z_1))$;
if there is a piece of slope 1 on $[z_1, z_4]$ of $\tilde{b}(p, q : t)$ **then**
  let p_{rise} be the only piece of slope 1 on the interval $[z_1, z_4]$ of $\tilde{b}(p, q : t)$;
  let $(z_2, \tilde{b}(p, q : z_2))$ and $(z_3, \tilde{b}(p, q : z_3))$ be the endpoints of p_{rise} with $z_2 \leq z_3$;
  let $y_2 \leftarrow z_2 - w(q, r)$ and $y_3 \leftarrow z_3 - w(q, r)$;
  traverse $\tilde{b}(r, s : t)$ from $(0, \tilde{b}(r, s : 0))$ to the right to $(\tilde{b}^{-1}(r, s : y_2), y_2)$;
  traverse $\tilde{b}(r, s : t)$ from $(e_2, \tilde{b}(r, s : e_2))$ to the left to $(\tilde{b}^{-1}(r, s : y_3), y_3)$;
  let L_1 be the remaining pieces on the interval $[\tilde{b}^{-1}(r, s : y_2), \tilde{b}^{-1}(r, s : y_3)]$ of $\tilde{b}(r, s : t)$;
  $L.push\_back((\tilde{b}^{-1}(r, s : y_2), 0))$;
  $L.append(L_1)$;
  $L.push\_back((e_2 - \tilde{b}^{-1}(r, s : y_3), 0))$;
else
  $L.push\_back((e_2, 0))$;
end if

if $z_4 \leq e_1$ **then**
  let L_2 be the remaining pieces on the interval $[z_4, e_1]$ of $\tilde{b}(p, q : t)$;
  $L.append(L_2)$;
end if
determine $point_{end}$ according to $point_{start}$ and L;

return $(b(p, s), point_{start}, point_{end}, L)$;

---

doi:10.6342/NTU202001303

To go a step further, given the records of $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ under the condition that $(p, ..., q, r, ..., s)$ is a subpath of $\mathcal{P}$, Procedure 5 constructs the record of $\tilde{b}(p, s : t)$ by compositing them in the time linear to the number of pieces in the record of $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ being removed during the process.

The main idea of Precedure 5 is according to the fact that $\tilde{b}(p, s : t) = \tilde{b}(p, q : w(q, r) + \tilde{b}(r, s : t))$. Besides, by Lemmas 4.2 and 4.3, the range of $w(q, r) + \tilde{b}(r, s : t)$ without regard to the last piece of slope 1 must be within length α, and be mapped to at most one rising piece in $\tilde{b}(p, q : t)$. Therefore, the record of $\tilde{b}(p, s : t)$ can be constructed efficiently. Below, we state 2 lemmas that analysis the correctness and the time complexity of Procedure 5.

Lemma 4.6. *Procedure 5 constructs the record of* $\tilde{b}(p, s : t)$ *correctly.*

Proof. Since $b(p, s) = \tilde{b}(p, q : w(q, r) + b(r, s))$ and $\tilde{b}(p, s : 0) = \tilde{b}(p, q : w(q, r) + \tilde{b}(r, s : 0))$, the value $b(p, s)$ and the coordinate of $point_{start} = (0, \tilde{b}(p, s : 0))$ are determined correctly. On the other hand, since $\tilde{b}(p, s : t) = \tilde{b}(p, q : w(q, r) + \tilde{b}(r, s : t))$, the slope of $\tilde{b}(p, s : t)$ on $[l, r]$ is 1 if and only if the slope of $\tilde{b}(r, s : t)$ on $[l, r]$ is 1 and the slope of $\tilde{b}(p, q : t)$ on $[w(q, r) + \tilde{b}(r, s : l), w(q, r) + \tilde{b}(r, s : r)]$ is 1.

Let $(e_1, \tilde{b}(p, q : e_1))$ and $(e_2, \tilde{b}(r, s : e_2))$ be the ending endpoints on the last piece of $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ respectively. The inequality $\tilde{b}(r, s : e_2) - \tilde{b}(r, s : 0) \leq \alpha$ holds due to Lemma 4.2. Therefore, according to Lemma 4.3, the function $\tilde{b}(p, q : t)$ on $[w(q, r) + \tilde{b}(r, s : 0), w(q, r) + \tilde{b}(r, s : e_2)]$ contains at most one piece of slope 1. If there is one piece p_{rise} of slope 1 on $[w(q, r) + \tilde{b}(r, s : 0), w(q, r) + \tilde{b}(r, s : e_2)]$ of $\tilde{b}(p, q : t)$ with endpoints $(z_2, \tilde{b}(p, q : z_2))$ and $(z_3, \tilde{b}(p, q : z_3))$ such that $z_2 = y_2 + w(q, r)$, $z_3 = y_3 + w(q, r)$, and $y_2 \leq y_3$, then we have the slope on $[0, \tilde{b}^{-1}(r, s : y_2)]$ of $\tilde{b}(p, s : t)$ is 0, and the slope on $[\tilde{b}^{-1}(r, s : y_3), e_2]$ of $\tilde{b}(p, s : t)$ is 0. Besides, the change trends of the function $\tilde{b}(p, s : t)$ on $[\tilde{b}^{-1}(r, s : y_2), \tilde{b}^{-1}(r, s : y_3)]$ and the function $\tilde{b}(p, q : t)$ on $[z_2, z_3]$ are the same. Otherwise, if there is no piece of slope 1 on $[w(q, r) + \tilde{b}(r, s : 0), w(q, r) + \tilde{b}(r, s : e_2)]$ of $\tilde{b}(p, q : t)$, then the slope on $[0, e_2]$ of $\tilde{b}(p, s : t)$ is 0. Clearly, Steps (10)-(22) implements the above arguments and constucts the list of pieces on $[0, e_2]$ of $\tilde{b}(p, s : t)$ correctly.

doi:10.6342/NTU202001303

As for $[e_2, \infty)$ of $\tilde{b}(p, s : t)$, since the slope on $[0, \infty)$ of $\tilde{b}(r, s : t)$ is 1, the change trends of the function $\tilde{b}(p, s : t)$ on $[e_2, \infty)$ and the function $\tilde{b}(p, q : t)$ on $[w(q, r) + \tilde{b}(r, s : e_2), \infty)$ are the same. Hence, Steps (24)-(27) constucts the list of pieces on $[e_2, \infty)$ of $\tilde{b}(p, s : t)$ correctly. It follows that the list of pieces L is constucted correctly, and the coordinate of $point_{end}$ is also determined correctly. $\square$

Lemma 4.7. *Procedure 5 constructs the record of $\tilde{b}(p, s : t)$ in $O(R)$ time, where R is the number of pieces in the records of $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ being removed during the process.*

Proof. Procedure 5 constructs the record of $\tilde{b}(p, s : t)$ by traversing the list of pieces in $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$. Note that those pieces in $\tilde{b}(p, q : t)$ and $\tilde{b}(r, s : t)$ being traversed in Steps (7), (14), and (15) will not appear in L. Thus, Steps (7), (14), and (15) takes exactly $O(R)$ time.

On the other hand, Steps (8)-(13), (16)-(19), and (24)-(27) can be done in $O(R)$ time since they can be done in the same time when traversing in Steps (7), (14), and (15). Besides, Steps (1)-(5) can be done in $O(1)$ time, and we can infer the differences of x-coordinate and y-coordinate of L in $O(1)$ time, implying that Step (28) can also be determined in $O(1)$ time. $\square$

4.3 Proofs of Lemmas 3.11, 3.12, and 3.13

Now, we get back to prove the correctness of Lemmas 3.11, 3.12, and 3.13, in order to ensure the $O(n)$ time complexity of Procedures 2 and 3.

4.3.1 Proof of Lemma 3.11

We calculate the broadcast time sequence $b(z_1, z_1), b(z_2, z_1), \ldots, b(z_m, z_1)$ in order. Since we have already determined all the values $b(x, T'_{x,z_1})$ where $T' = T(z_1, z_1))$ and x is the neighbor of z_1 in T', the value $b(z_1, z_1)$ can be determined in $O(deg(z_1))$ time by Lemma 1.3.

doi:10.6342/NTU202001303

Assume that we have determined all the value $b(z_j, z_1)$ with $1 \leq j < i$, and we are now going to determine $b(z_i, z_1)$ for some $i \geq 2$. Without loss of generousity, let $T' = T(z_1, z_i)$. Note that the neighbor x of z_i in T' is either z_{i-1} or some vertex in T_{z_i}. We have already determined all the values $b(x, T'_{x,z_i}) = b(x, T_{x,z_i})$ when x is the neighbor of z_i in T_{z_i}. On the other hand, when $x = z_{i-1}$ and $2 \leq i \leq m$, $b(x, T'_{x,z_i}) = b(z_{i-1}, z_1)$ is determined in the previous term of the sequence. Therefore, Lemma 1.3 implies that the value $b(z_i, z_1)$ can be determined in $O(deg(z_i))$ time. By repeatly using this procedure, we can calculate all the broadcast time $b(z_1, z_1), b(z_2, z_1), \ldots, b(z_m, z_1)$ in $O(\sum\limits_{i=1}^{m} deg(z_i))$ time.

4.3.2 Proof of Lemma 3.12

We calculate the broadcast time sequence $b(z_1, z_1), b(z_1, z_2), \ldots, b(z_1, z_m)$ in order. Note that the record of $\tilde{b}(u, v : t)$ includes the value $b(u, v)$. Using Procedure 4, the records of $\tilde{b}(z_1, z_1 : t), \tilde{b}(z_1, z_1 : t)$ can be constructed in $O(deg(z_1))$ time, and the value $b(z_1, z_1)$ is obtained in the same time.

Assume that we have constructed the record of $\tilde{b}(z_1, z_{i-1} : t)$ for some $i \geq 2$, by compositing the records of $\tilde{b}(z_1, z_{i-1} : t)$ and $\tilde{b}(z_i, z_i : t)$ using Procedure 5, we can construct the record of $\tilde{b}(z_1, z_i : t)$ in $O(r_i)$ time where r_i is the number of pieces being removed during the process. Note that the record of $\tilde{b}(z_i, z_i : t)$ can be consturcted in $O(deg(z_i))$ time by Lemma 4.5, implying that the number of pieces in the record of $\tilde{b}(z_i, z_i : t)$ is also bounded by $O(deg(z_i))$. Besides, all the pieces being removed are actually correspond to some pieces appearing in the records of $\tilde{b}(z_1, z_1 : t)$, $\tilde{b}(z_2, z_2 : t)$, ..., $\tilde{b}(z_m, z_m : t)$. Therefore, the total running time of constucting these records one by one will be $O(\sum\limits_{i=2}^{m} r_i) = O(\sum\limits_{i=1}^{m} deg(z_i))$, and the values $b(z_1, z_2)$, $b(z_1, z_3)$, ..., $b(z_1, z_m)$ are obtained at the same time.

4.3.3 Proof of Lemma 3.13

Without loss of generousity, we assume that the above broadcast time sequence covers each $\mathcal{P}$-broadcast function $\tilde{b}(z_1, z_1 : t)$, $\tilde{b}(z_2, z_1 : t)$, ..., $\tilde{b}(z_m, z_1 : t)$. Besides, we assume

doi:10.6342/NTU202001303

that the input values of $\tilde{b}(z_i, z_1 : t)$ is constrained to $[l_i, r_i]$ for $1 \leq i \leq m$, and the equation $l_i = r_{i+1}$ holds for $1 \leq i \leq m - 1$. By the above assumptions, the process of determining the broadcast time sequence can be split into two parts. The first part is to traverse $\tilde{b}(z_i, z_1 : t)$ from $t = r_i$ to $t = l_i$ and determine the broadcast time in the sequence, and the second part is to determine $\tilde{b}(z_{i+1}, z_1 : r_{i+1})$ from $\tilde{b}(z_i, z_1 : l_i)$ by a slightly modification of Procedure 5.

For each $\tilde{b}(z_i, z_1 : t)$ with $1 \leq i \leq m$, we use the pointer p_i to point to the location in $[l_i, r_i]$ of its record in order to determine the broadcast time appearing in the sequence. For the first part, we move the pointer p_i from r_i to l_i in the record of $\tilde{b}(z_i, z_1 : t)$. Note that if a piece of $\tilde{b}(z_i, z_1 : t)$ is traversed, it will not be traversed again in the following process. Besides, all the pieces traversed are actually correspond to some pieces appearing in the records of $\tilde{b}(z_1, z_1 : t)$, $\tilde{b}(z_2, z_2 : t)$, ..., $\tilde{b}(z_m, z_m : t)$. Therefore, since the number of pieces in the record of $\tilde{b}(z_i, z_i : t)$ is bounded by $O(deg(z_i))$ by Lemma 4.5, the first part takes at most $O(\sum_{i=1}^{m} deg(z_i))$ time. For the second part, one can see that value $\tilde{b}(z_{i+1}, z_1 : r_{i+1})$ can be infered from $\tilde{b}(z_i, z_1 : l_i)$ and the location pointed by p_{i+1} can be also inferred by p_i by slightly modifying Procedure 5. It follows that the second part takes at most $O(\sum_{i=1}^{m} deg(z_i) + n)$ time. Therefore, we can calculate the broadcast time sequence $\tilde{b}(z_{a_1}, z_1 : v_1)$, $\tilde{b}(z_{a_2}, z_1 : v_2)$, ... , $\tilde{b}(z_{a_n}, z_1 : v_n)$ in the total of $O(\sum_{i=1}^{m} deg(z_i) + n)$ time.

doi:10.6342/NTU202001303

Chapter 5

Concluding Remarks

In this thesis, we proposed an $O(n)$ time algorithm to solve the broadcasting 2-center problem in weighted trees under the postal model. The result is optimal for finding broadcast 2-centers. We observe that the problem can be solved by finding out the essential edge, and prove that the candidate path $\mathcal{P}$ contains an essential edge. To find the essential edge on $\mathcal{P}$, we determine the broadcast center and calculate 1-center broadcast time of each $\mathcal{P}$-subtree.

The main challenge in this problem is to determine those broadcast time sequences in Lemmas 3.11, 3.12, and 3.13 in an efficient way. As the adjacent $\mathcal{P}$-subtrees share much in common, we use the concept of $\mathcal{P}$-broadcast function to store information and query each broadcast time without recomputing every time. As a consequence, those broadcast time sequences can all be determined in linear time as desired.

doi:10.6342/NTU202001303

Bibliography

[1] P. J. Slater, E. J. Cockayne, and S. T. Hedetniemi, “Information dissemination in trees,” *SIAM Journal on Computing*, vol. 10, no. 4, pp. 692–701, 1981.

[2] M. Elkin and G. Kortsarz, “Sublogarithmic approximation for telephone multicast,” *Journal of Computer and System Sciences*, vol. 72, no. 4, pp. 648–659, 2006.

[3] U. Feige, D. Peleg, P. Raghavan, and E. Upfal, “Randomized broadcast in networks,” *Random Structures Algorithms*, vol. 1, no. 4, pp. 447–460, 1990.

[4] P. Fraigniaud and S. Vial1, “Approximation algorithms for broadcasting and gossiping,” *Journal of Parallel and Distributed Computing*, vol. 43, no. 1, pp. 47–55, 1997.

[5] G. Kortsarz and D. Peleg, “Approximation algorithms for minimum-time broadcast,” *SIAM Journal on Discrete Mathematics*, vol. 8, no. 3, pp. 401–427, 1995.

[6] A. Jakoby, R. Reischuk, and C. Schindelhauer, “The complexity of broadcasting in planar and decomposable graphs,” *Discrete Applied Mathematics*, vol. 83, no. 1-3, pp. 179–206, 1998.

[7] A. Dessmark, A. Lingas, H. Olsson, and H. Yamamoto, “Optimal broadcasting in almost trees and partial k-trees,” *Lecture Notes in Computer Science 1373: STACS 98*, pp. 432–443, 1998.

[8] H. A. Harutyunyan and E. Maraachlian, “Broadcasting in fully connected trees,” *Proceedings of the 15th International Conference on Parallel and Distributed Systems*, pp. 740–745, 2009.

doi:10.6342/NTU202001303

[9] H. A. Harutyunyan and E. Maraachlian, “On broadcasting in unicyclic graphs,” *Journal of Combinatorial Optimization*, vol. 16, pp. 307–332, 2008.

[10] H. A. Harutyunyan and P. Taslakian, “Orderly broadcasting in a 2D torus,” *Proceedings of the 8th International Conference on Information Visualisation*, pp. 370–375, 2004.

[11] R. Beier and J. F. Sibeyn, “A powerful heuristic for telephone gossiping,” *Proceedings of the 7th International Colloquium on Structural Information and Communication Complexity*, pp. 17–36, 2000.

[12] H. A. Harutyunyan and C. Jimborean, “New heuristic for message broadcasting in networks,” *Proceedings of the 28th IEEE International Conference on Advanced Information Networking and Applications*, pp. 517–524, 2014.

[13] H. A. Harutyunyan and B. Shao, “An efficient heuristic for broadcasting in networks,” *Journal of Parallel and Distributed Computing*, vol. 66, no. 1, pp. 68–76, 2006.

[14] P. Scheuermann and G. Wu, “Heuristic algorithms for broadcasting in point-to-point computer networks,” *IEEE Transactions on Computers*, vol. C-33, no. 9, pp. 804–811, 1984.

[15] A. de Sousa, G. Gallo, S. Gutierrez, F. Robledo, P. Rodríguez-Bocca, and P. Romero, “Heuristics for the minimum broadcast time,” *Electronic Notes in Discrete Mathematics*, vol. 69, pp. 165–172, 2018.

[16] J.-M. Koh and D.-W. Tcha, “Information dissemination in trees with nonuniform edge transmission times,” *IEEE Transactions on Computers*, vol. 40, no. 10, pp. 1174–1177, 1991.

[17] H. A. Harutyunyan, A. L. Liestman, and B. Shao, “A linear algorithm for finding the k-broadcast center of a tree,” *Networks*, vol. 53, no. 3, pp. 287–292, 2009.

doi:10.6342/NTU202001303

[18] C.-H. Tsou, G.-H. Chen, H.-I. Yu, and C.-C. Lin, “The broadcast median problem in heterogeneous postal model,” *Journal of Combinatorial Optimization*, vol. 25, no. 4, pp. 602–616, 2013.

[19] Y.-H. Su, C.-C. Lin, and D. T. Lee, “Broadcasting in weighted trees under the postal model,” *Theoretical Computer Science*, vol. 621, pp. 73–81, 2016.

[20] C.-H. Tsou, G.-H. Chen, and C.-C. Lin, “Broadcasting in heterogeneous tree networks with uncertainty,” *Lecture Notes in Computer Science 7074: Algorithms and computation*, pp. 200–209, 2011.

doi:10.6342/NTU202001303