Computation of Broadcasting Multiple Messages in a Positive Weighted Tree

A. Averbuch Y. Roditty B. Shoham
Department of Computer Science
School of Mathematical Sciences
Tel Aviv University
Ramat Aviv, Tel Aviv 69978
Israel

e-mail: barack@math.tau.ac.il

Abstract

In this paper we present algorithms for locating the vertices in a tree of n vertices of positive edge-weighted tree and a positive vertex-weighted tree from which we broadcast multiple messages in a minimum cost. Their complexity is $O(n^2 \log n)$. It improves a direct recursive approach which gives $O(n^3)$. In case where all the weights are equal to one the complexity is $O(n)$.

1 Introduction

Efficient broadcasting is a key component in achieving high performance (throughput) from parallel and distributed processing.

The motivation for this work was triggered by our interest in investigating the problem how to perform optimal query on distributed database on diverse MIMD multiprocessor architectures [1]. There we investigated how to schedule and evaluate query with minimal cost.

Broadcasting has become popular in recent years with the introduction of new, very fast networks and their extensive use by parallel and distributed systems. Utilizing efficiently the underlying network in a massively parallel/distributed system is a major challenge. Recently, new demanding applications of broadcast and group communication have evolved. Video conferencing and database processing, wide scale distribution of audio, video and data via commercial television digital networks (video on demand), multi-user games, on line multimedia databases and remote tutoring are typical examples which are very demanding in terms of bandwidth requirements.

Edge/Vertex-weighted topology is applicable when we want to rate different locations in heterogeneous distributed network. Edge/Vertex-weighted topology can also resembles transmissions to a subset of the processors called hereinafter *multicast*, where we broadcast to vertices such that their weight is greater than a given threshold. Moreover, this topology enables us to take into consideration the delay of processing a message prior to its transmission and the edges capacity. Therefore, having efficient algorithms for broadcasting in positive edge/vertex weighted-tree are of great use in parallel/distributed networks.

We define broadcasting from an *originator(s)* source to be the process of passing one (many) unit(s) of information from that source to a set of destinations which are connected via a network.

This is accomplished by a series of transitions over the network. The messages are distributed over the network and spread using the communication network, where each node transmits a message to its neighbors upon receiving it regardless to the activity done in other nodes (beside for the node that receives the message that should be idle). The broadcasting problem is usually described by the following rules:

1. A vertex in the network may send a message to an adjacent vertex only.
2. Time is discrete. At a given time each vertex will do exactly one of the following:
   (a) receive a message,
   (b) send a message to one neighbor,
   (c) be idle.

Eventually, broadcasting should be logically viewed as a many-to-many communication.

Formally, we can view the communication network as a finite connected undirected graph where the set of vertices stands for processors and each edge which connects two vertices stands for a direct communication link between these vertices. Then, we define broadcasting from a vertex v (*the originator*) as transmission of a message (messages) from v to every vertex in $V \setminus \{v\}$ using the above rules. This problem, that was introduced and solved in [15], is a variation of the gossiping problem [8].

The problem of broadcasting in a general graph is NP-complete [7]. On the other hand, on a tree with equal weights it is known to be linear [15]. In addition, the cost to broadcast a message from any other vertex, say v, was shown to be the summation of the minimal broadcasting costs and the distance from v to a vertex in the set of vertices from which the broadcast cost is minimal. In other topologies except trees there are known results in a two and three dimensional grid [16], complete graph and hypercubes (see [5], [6] and [10]). Recently, planar graphs were treated as well [9]. A conjecture concerning minimum broadcasting time starting from a given vertex in a $d-$dimensional grid ($d \geq 3$), was posed in [16]. This conjecture was recently validated in [12]. The concept of *time-relaxed* broadcasting was introduced in [13] and results obtained for trees and general graphs in [13, 14] and [2, 3].

The main goal of this paper is generalization of the result in [15] by presenting algorithms which enable efficient broadcasting of multiple messages in parallel/distributed multiprocessors on a positive weighted tree topology.

Multicast communication which has been recognized as an important and useful mechanism since the mid-seventies can be defined as a message(s) transmission mechanism that delivers a message(s) from a single (many) source(s) to a set of destinations (many-to-many communication)[11]. Broadcast (many-to-all communication) is a special forms of multicast.

The goals of the algorithms:

1. To identify the set of vertices from which the *cost*[1] to broadcast is minimal. This set is called hereafter *broadcast center.*
2. To evaluate the *cost*[1] to broadcast messages from any vertex in the tree, and
3. To evaluate the *cost*[1] to broadcast $m \geq 1$ messages from any vertex in the tree.

[1]The notion of *cost* is defined in section 2.1 for the edge-weighted tree and in section 3.1 for the vertex-weighted tree.

Denote by T the tree the tree and n its size. Our algorithms have time complexity of $O(n^2 \log n)$. In the case where all the weights are equal, the time complexity is $O(n)$ as in [15]. If we calculate directly the cost in each vertex we get $O(n^2)$ so that the total cost is $O(n^3)$.

The paper is organized as follows: The algorithm that deals with weights which are assigned to edges is presented in section 2. In section 3 we present the algorithm to do the same when the weights are assigned to vertices.

2 Broadcasting in Edge-Weighted Tree

2.1 Formal Definition of the Cost

For a given graph $G = G(V, E)$ and a weight $w(e)$, $e \in E(G)$, we define the *broadcast number* of $v \in V(G)$, denoted by $b(v, G)$, as the minimum *cost* that is required to broadcast one message from v. Let $e = (u, v)$ be an edge and let $t(e)$ be the time a message is transmitted from u to v. The value $w(e) \times t(e)$ presents in some sense the edge priority related to the arrival time of the message.

The *cost* to broadcast a message is calculated by one of the following measures:

2.1.1 $\max_{e \in E}\{w(e) \times t(e)\}$.

This cost function represents the maximum value of the edge weight related to the message arrival time. Its aim is to minimize the maximum time a vertex "waits" for a message related to its priority to get it.

2.1.2 $\frac{\sum_{e \in E}\{w(e) \times t(e)\}}{|E|}$.

This cost function represents the average value of the edge weight related to the message arrival time. Its aim is to minimize the average time a vertex "waits" for a message related to its priority to get it.

The *broadcast center* of G, denoted by $BC(G)$, is defined, as in [15], to be the set of all vertices having minimum broadcasting number. The broadcast number of G is $b(G) = b(u, G)$ where $u \in BC(G)$.

2.2 Locating the Broadcast Center in Edge-Weighted Tree

The proposed algorithm for locating the broadcast center in an edge-weighted tree is based upon the following:

Denote $T_v^{(u,v)}$ to be the tree in the forest $T \setminus \{(u, v)\}$ that contains v. Define $f(T, v, t)$ to be the cost function that expresses the minimum cost to broadcast from v in T when broadcast starts at time t. The edge (v, u_i) is denoted $e_v^{u_i}$

Observation I:

For a given vertex $v \in V(T)$, let $u_1, u_2, \ldots, u_k$ be the set of neighbors of v. The value of $f(T, v, t)$ is computed based upon the value of the cost function of its adjacent vertices and the weight of the edges connecting v to them. Namely, it is computed using $f(T_{u_i}^{(v,u_i)}, u_i, t)$ and $w(e_v^{u_i})$, $i = 1, \ldots, k$. The order of transmitting the message from v to u_i, $i = 1, \ldots, k$ is as follows: assume, without loss of generality, that $f(T_{u_i}^{(v,u_i)} \cup e_v^{u_i}, v, t) \geq f(T_{u_{i+1}}^{(v,u_{i+1})} \cup e_v^{u_{i+1}}, v, t)$. Then, the optimal cost to broadcast a message from v is achieved by sending the message to u_1 first then to u_2 and so on until it reaches u_k. ■

Example:

The idea is to assign the value of the cost function to each vertex starting from the leaves using the value of the cost function of its adjacent vertices.

Assume $u \in V(T)$ and v is its father. Then, starting from the leaves of T, we assign to each $u \in V(T)$ the value $f(T_u^{(v,u)}, u, t)$. The cost function assigns zero to a leaf since the cost to broadcast in a subtree that includes only one vertex (the leaf) is zero. In Fig. 1A there is an example for the cost measure defined in 2.1.1. The values of the cost function in the vertices adjacent to the leaves are:

$$f(T_{v_1}^{(v_0,v_1)}, v_1, t) = \max\{f(T_{v_2}^{(v_1,v_2)}, v_2, t+1),\ w(e_{v_1}^{v_2}) \times (t+1)\} = \max\{0,\ 2 \times (t+1)\} = 2t+2$$

and,

$$f(T_{v_3}^{(v_0,v_3)}, v_3, t) = \max\{f(T_{v_4}^{(v_3,v_4)}, v_4, t+1),\ w(e_{v_3}^{v_4}) \times (t+1)\} = \max\{0,\ 1 \times (t+1)\} = t+1.$$

Notice that we take the maximum between the cost to broadcast in the subtree $T_{v_2}^{(v_1,v_2)}$ or $T_{v_4}^{(v_3,v_4)}$ and the cost to transmit the message over the edge (v_1, v_2) or (v_3, v_4), respectively.

We transmit the message from v_0 to v_3 and then to v_1 since

$$f(T_{v_1}^{(v_0,v_1)} \cup e_{v_0}^{v_1}, v_0, t) = \max\{w(e_{v_o}^{v_1}) \times (t+1),\ f(T_{v_1}^{(v_0,v_1)}, v_1, t+1)\} = \max\{1(t+1),\ 2(t+1)+2)\} = 2(t+2)$$

and

$$f(T_{v_3}^{(v_0,v_3)} \cup e_{v_0}^{v_3}, v_0, t) = \max\{w(e_{v_0}^{v_3}) \times (t+1),\ f(T_{v_3}^{(v_0,v_3)}, v_3, t+1)\} = \max\{100(t+1),\ t+2\} = 100(t+1)$$

Therefore,

$$f(T_{v_3}^{(v_0,v_3)} \cup e_{v_0}^{v_3}, v_0, t) > f(T_{v_1}^{(v_0,v_1)} \cup e_{v_0}^{v_1}, v_0, t).$$

In the above we calculated the cost to broadcast a message from v_0 to the trees rooted by only one of its adjacent and found that the optimal transmission order is by sending the message first to v_2 and then to v_1. Therefore, $f(T, v_0, t) =$

$$\max\left\{\max\left(f(T_{v_3}^{(v_0,v_3)}, v_3, t+1), 100 \times (t+1)\right), \max\left(f(T_{v_1}^{(v_0,v_1)}, v_1, t+2), 1 \times (t+2)\right)\right\} =$$

$$\max\{\max((t+1)+1, 100 \times (t+1)) \max(2(t+2)+2, 1 \times (t+2))\}.$$

Since v_0 is the originator we have $t = 0$ and the cost to broadcast from v_0 is

$$f(T, v_0, 0) = \max\{\max(2, 100), \max(6, 2)\} = 100.$$

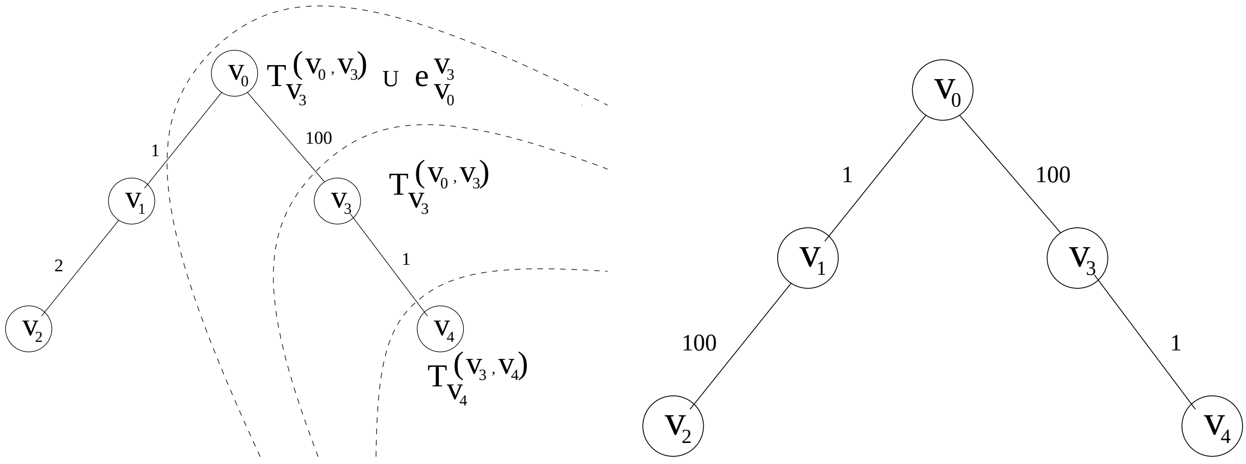


Figure 1: Edge-weighted tree.

In Fig. 1B there is an example for the cost measure defined in 2.1.1. The values of the cost function in the vertices adjacent to the leaves are:

$$f(T_{v_1}^{(v_0,v_1)}, v_1, t) = 100t + 100$$

and,

$$f(T_{v_3}^{(v_0,v_3)}, v_3, t) = \max\{0,\ 1 \times (t+1)\} = t + 1.$$

We transmit the message from v_0 to v_1 and then to v_3 since

$$f(T_{v_1}^{(v_0,v_1)} \cup e_{v_0}^{v_1}, v_0, t) = 100(t+1) + 100 = 100t + 200$$

and

$$f(T_{v_3}^{(v_0,v_3)} \cup e_{v_0}^{v_3}, v_0, t) = 100(t+1) = 100t + 100$$

Therefore,

$$f(T_{v_1}^{(v_0,v_1)} \cup e_{v_0}^{v_1}, v_0, t) > f(T_{v_3}^{(v_0,v_3)} \cup e_{v_0}^{v_3}, v_0, t)$$

and

$$f(T, v_0, t) = \max\left\{\max\left(1(t+1), 100(t+1) + 100\right) \max\left(100(t+2), (t+2) + 1\right)\right\}.$$

Since v_0 is the originator we have $t = 0$ and the cost to broadcast from v_0 is

$$f(T, v_0, 0) = \max\left\{\max\left(1, 200\right), \max\left(200, 3\right)\right\} = 200.$$

Remark:

We need to keep track of all possible values of the time t in which the vertex receives a message since t is unknown while the algorithm in progress and the cost function $f(T, v, t)$ depends upon t. ■

In the following we assume that "cost" means either 2.1.1 or 2.1.2.

We use the following notations:

U	the set of labeled vertices that were removed from T.
$L_{\bar{U}}$	the set of labeled vertices that were not removed from T.
T_U	the tree $T \setminus U$.
$t_{t_1}^{t_2}$	time interval from time t_1 to t_2. The interval t_0^{∞} stands for any time.
$T_v^{(u,v)}$	the tree in the forest $T \setminus \{(u,v)\}$ that contains v.
$f(T,v,t)$	cost function that expresses the minimum cost to broadcast from v in T when broadcast starts at time t.
$W(v)$	ordered list of pair $(t_{t_1}^{t_2}, f(T,v,t))$ where T is a given tree and $t \in t_{t_1}^{t_2}$.
v_{op}	vertex from which broadcast is optimal.
$Center(T)$	set of vertices v_{op} that was chosen by the algorithm.
$Number(T)$	the value of $f(T, v_{op}, t)$ in the time interval $t_0^{t_2}$ at time $t = 0$.

In the following algorithm, *Broadcast I*, we compute in each step the cost to broadcast from a vertex to its sons, starting from the leaves. This algorithm is a generalization of the algorithm presented in [15].

Algorithm **Broadcast I**:

1 (Label the leaves of T with cost 0 for t_0^∞)

Let U be the set of leaves of T;

for each $u \in U$ **do**:

$W(u) \leftarrow \{(t_0^\infty, 0)\}$;

end for

$T_U \leftarrow T \setminus U$.

2 (Label the leaves of T_U.)

Let $L_{\bar{U}}$ be the set of leaves in T_U;

for each $u \in L_{\bar{U}}$ **do**:

Let v be the father of u in T_U.

2.1 **for each** $t_{t_1}^{t_2}$ which gives different value of $f(T_u^{(v,u)}, u, t),\ t \in t_{t_1}^{t_2}$, **do**:

$W(u) \leftarrow W(u) \cup \{(t_{t_1}^{t_2}, f(T_u^{v,u)}, u, t)\}$;

end for each

end for

3 (Select the next vertex to be deleted and the next vertex to be labeled, until there is only one vertex left)

While $|V(T_U)| > 1$ **do**:

3.1 (select the next vertex v which is the father of a vertex u in T_U)

Let $u \in L_{\bar{U}}$ where $f(T_u^{(v,u)}, u, t) = \min_{u \in L_{\bar{U}}}\{f(T_u^{(v,u)}, u, 0)\}$

where v is the father of u in T_U.

3.2 (Delete u from $L_{\bar{U}}$ and T_U and add it to U)

$L_{\bar{U}} \leftarrow L_{\bar{U}} \setminus \{u\}$;

$U \leftarrow U \cup \{u\}$;

$T_U \leftarrow T_U \setminus \{u\}$;

3.3 (Label the next vertex.)

if v is a leaf of T_U **then**

Let w be the father of v in T_U

3.4 **for each** $t_{t_1}^{t_2}$ which gives different value of $f(T_v^{(u,v)}, v, t),\ t \in t_{t_1}^{t_2}$, **do**:

$W(v) \leftarrow W(v) \cup \{(t_{t_1}^{t_2}, f(T_v^{(u,v)}, v, t))\}$;

end for each

$L_{\bar{U}} \leftarrow L_{\bar{U}} \cup \{v\}$;

end if

end while

4 (There is one vertex left in T_U.)

$v_{op} \leftarrow V(T_U)$;
$Number(T) \leftarrow f(T, v_{op}, 0)$.
$Center(T) \leftarrow \{v_{op}\}$;

4.1 (pick vertices $u \in V(T)$ where $b(u,T) = b(v_{op}, T)$)
Let $u_1, u_2, \ldots, u_k$ be the neighbors of v in T.
for each u_i, $1 \leq i \leq k$ **do**:
$W(u_i) \leftarrow \{(t_o^z, f(T, u_i, t)\}$ where $z \geq 0$;
if $f(T, u_i, 0) = Number(T)$ **then**:
$Center(T) \leftarrow Center(T) \cup \{u_i\}$;
end if
end for each

Let v be any vertex in V with $u_1, u_2, \ldots, u_k$ as its neighbors. Then, as was done in the previous example we compute the cost functions according to the following:

1. The cost function $C(T, v, t)$ of v computed according to 2.1.1 in the following way:

$$C(T, v, t) = \max_{1 \leq i \leq k} \left\{ \max \left\{ C(T_{u_i}^{(v,u_i)}, u_i, t+i), w(e_v^{u_i}) \times (t+i) \right\} \right\},$$

where the neighbors of v are ordered such that $C(T_{u_i}^{(v,u_i)} \cup e_v^{u_i}, v, t) \geq C(T_{u_{i+1}}^{(v,u_{i+1})} \cup e_v^{u_{i+1}}, v, t)$, $1 \leq i \leq k-1$ and $e_v^{u_i} = (v, u_i)$. For a leaf v in the tree T, $C(T, v, t) = 0$.

2. For the cost function $AC(T, v, t)$ of v computed according to 2.1.2, we keep the value of the cost function of each vertex of the tree, and the number of vertices in which this function was calculated. The cost function $AC(T, v, t)$, is computed in the following way:

$$AC(T, v, t) = \frac{\sum_{i=1}^{k} \left(AC(T_{u_i}^{(v,u_i)}, u_i, t+i) + w(e_v^{u_i}) \times (t+i) \right)}{\sum_{i=1}^{k} n_{u_i} + 1},$$

where the neighbors of v are ordered as in 1 and n_{u_i} is the number of vertices in $T_{u_i}^{(v,u_i)}$. For a leaf v in the tree T, $AC(T, v, t) = 0$ and $n_v = 1$.

Remark 1:

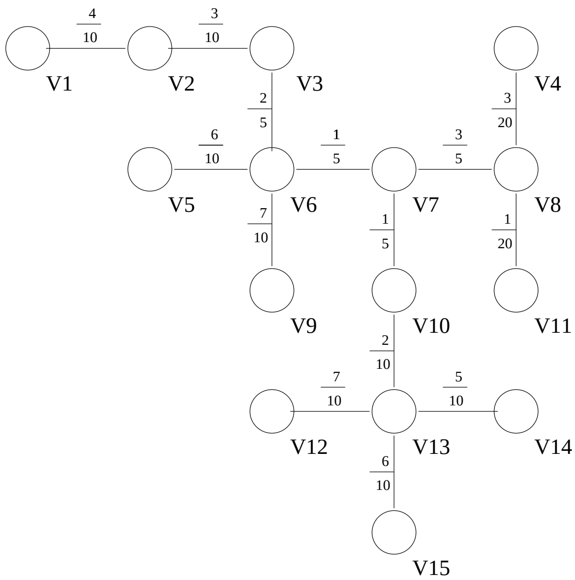
Algorithm *Broadcast I* we assign the functions AC or C instead of f according to the required cost measurement type (2.1.1. or 2.1.2.).

Remark 2:

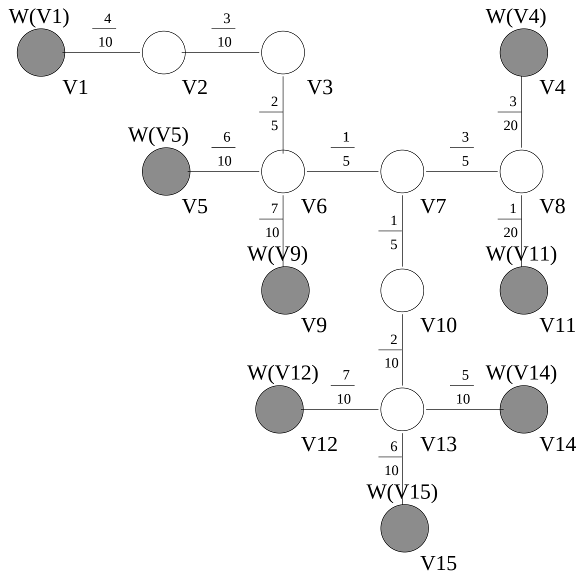
For the case where $\{w(e_1) = w(e_2),\ \forall\ e_1, e_2 \in E\}$ we get only one time interval (t_0^{∞}). Therefore, steps 2.1 and 3.4 of the algorithm are executed only once and this algorithm coincides with the algorithm presented in [15] which can be viewed as a special case of the algorithm *Broadcast I*. ■

Example:

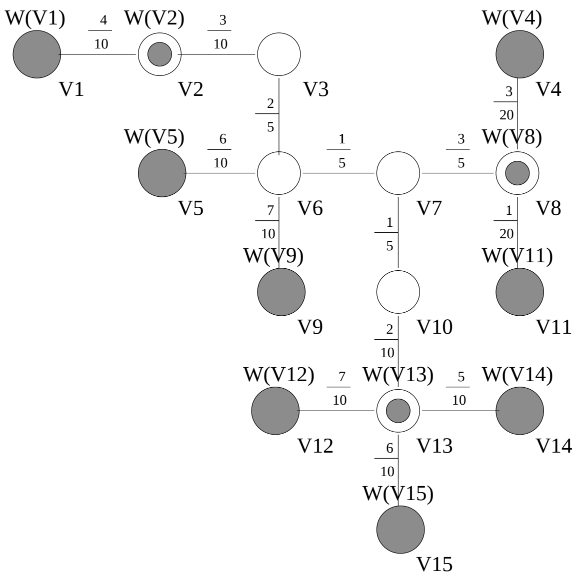
A detailed example how to find the *broadcast center* for an edge weighted tree is presented in Fig. 2 and table 1 for the cost function $C(T, v, t)$. Fig. 2 presents the steps of the algorithm where the white vertices are the vertices in T_U, the gray vertices are in U and the partial gray vertices are in $L_{\bar{U}}$. In table 1 we present the values of $W(v)$, $v \in V(T)$, and the transmission order from v to its adjacent vertices, when the time is recorded as the time when the message was received. The vertices are ordered according to the way they are accessed by the algorithm (see column 1 in table 1). Notice that v_7 is the last vertex that remains in T_U. Therefore, it belongs to the *broadcast center*. v_7 was visited twice. The value of $Number(T)$ is the value of the second coordinate of $W(v_7)$ at time $t = 0$. Hence, $(\frac{4}{10} \times (t+7) \mid t = 0) = \frac{28}{10}$ and $BC(T) = \{v_7, v_{10}\}$. The vertex v_{10} is added to $BC(T)$ in step 4 of the algorithm.



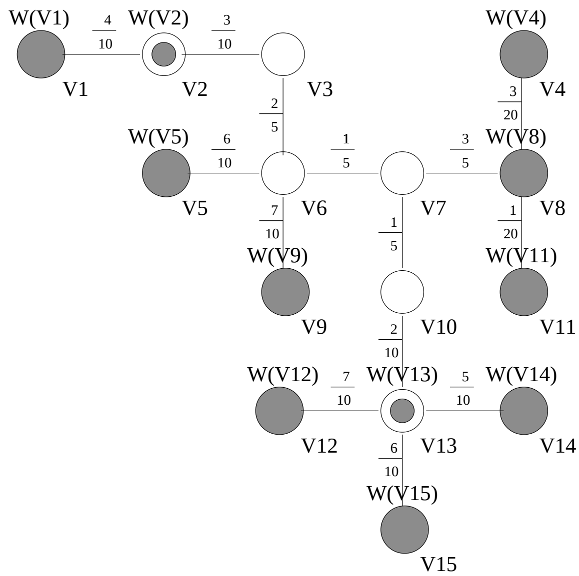
A: Initial configuration of the edge weighted tree.



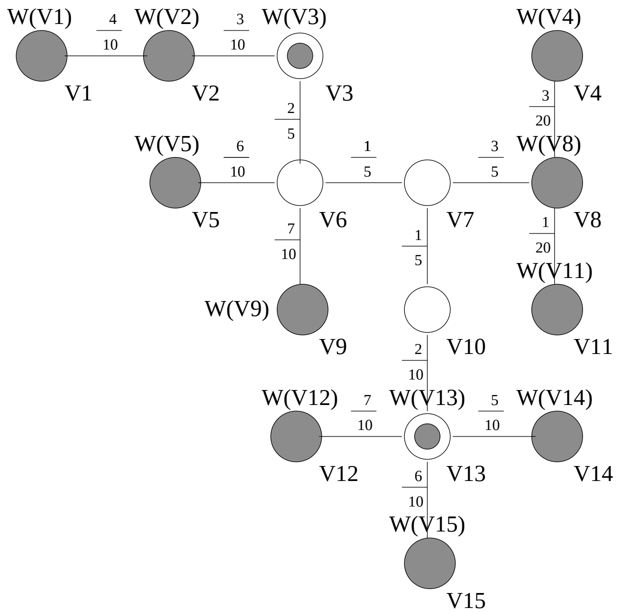
B: The tree after step 1.



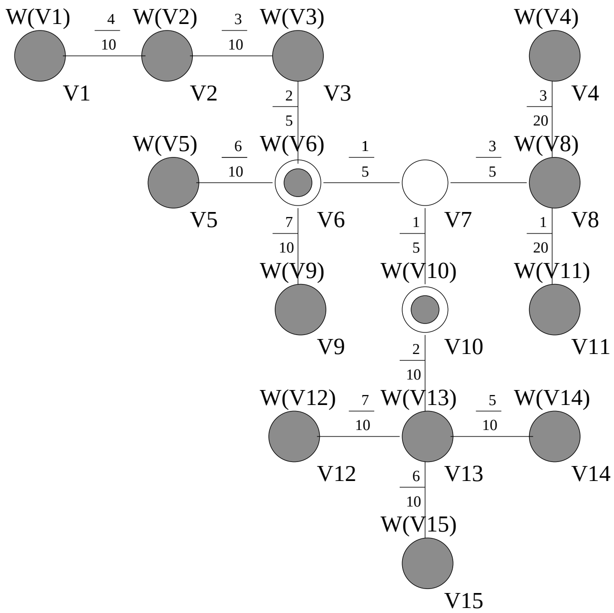
C: The tree after step 2



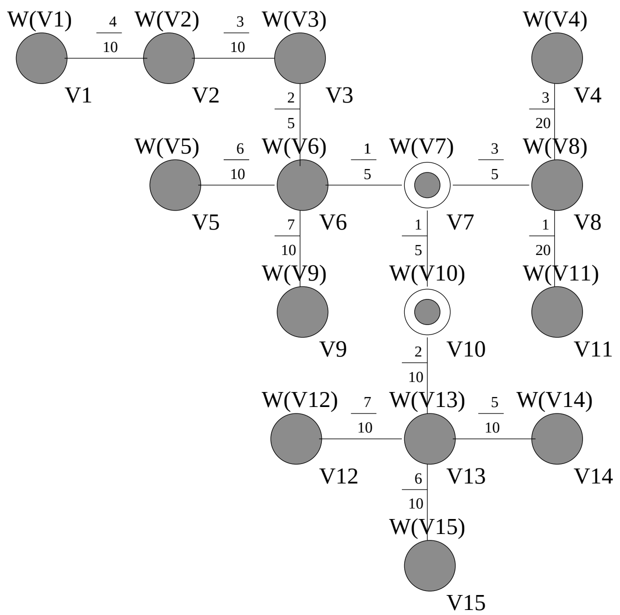
D: The tree after the first iteration of step 3.



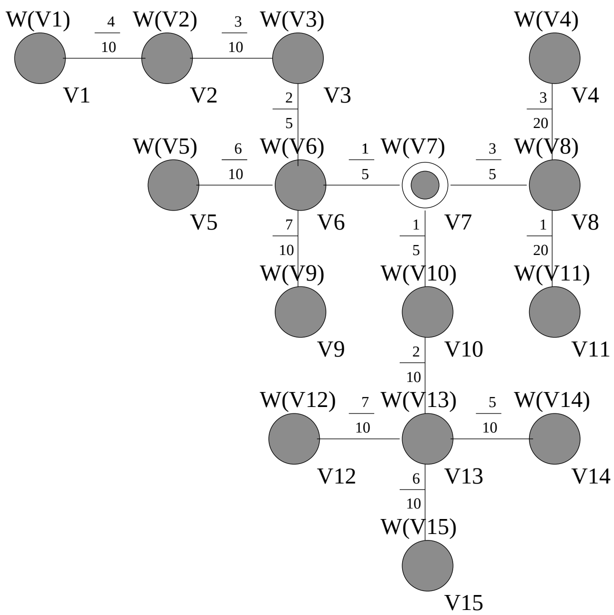
E: The tree after the second iteration of step 3.



F: The tree after the third and fourth iteration of step 3.



G: The tree after the fifth iteration of step 3.



F: The tree after the sixth iteration of step 3

Figure 2: Demonstration how to locate the *broadcast center* in edge-weighted tree

Seq.	Alg. step No.-	Vertex	$W(v_i)$	transmission order
1	1	v_1	$\{t_0^\infty, 0)\}$	
2	1	v_5	$\{(t_0^\infty, 0)\}$	
3	1	v_9	$\{(t_0^\infty, 0)\}$	
4	1	v_{12}	$\{(t_0^\infty, 0)\}$	
5	1	v_{15}	$\{(t_0^\infty, 0)\}$	
6	1	v_{14}	$\{(t_0^\infty, 0)\}$	
7	1	v_{11}	$\{(t_0^\infty, 0)\}$	
8	1	v_4	$\{(t_0^\infty, 0)\}$	
9	2	v_2	$\{(t_0^\infty, \frac{4}{10} \times (t+1))\}$	(t_0^∞) : v_1
10	2	v_{13}	$\{(t_0^3, \frac{5}{10} \times (t+3))$ $(t_4^5, \frac{6}{10} \times (t+2))$ $(t_6^\infty, \frac{7}{10} \times (t+1))\}$	(t_0^∞) : v_{12}, v_{15}, v_{14}
11	2	v_8	$\{(t_0^\infty, \frac{3}{20} \times (t+1))\}$	(t_0^∞) : v_4, v_{11}
12	3.4	v_3	$\{(t_0^\infty, \frac{4}{10} \times (t+2))\}$	(t_0^∞) : v_2
13	3.4	v_6	$\{(t_0^1, \frac{6}{10} \times (t+3))$ $(t_2^4, \frac{4}{10} \times (t+5))$ $(t_5^5, \frac{6}{10} \times (t+2))$ $(t_6^\infty, \frac{7}{10} \times (t+1))\}$	$(t_0^1\ )$: v_3, v_9, v_5 (t_2^∞) : v_9, v_5, v_3
14	3.4	v_{10}	$\{(t_0^3, \frac{5}{10} \times (t+4))$ $(t_4^4, \frac{6}{10} \times (t+3))$ $(t_5^\infty, \frac{7}{10} \times (t+2))\}$	(t_0^∞) : v_{13}
15	3.4	v_7	$\{(t_0^0, \frac{6}{10} \times (t+4))$ $(t_1^3, \frac{4}{10} \times (t+6))$ $(t_4^4, \frac{6}{10} \times (t+3))$ $(t_5^\infty, \frac{7}{10} \times (t+2))\}$	(t_0^∞) : v_6, v_8
16	3.4	v_7	$\{(t_0^2, \frac{4}{10} \times (t+7))$ $(t_3^3, \frac{6}{10} \times (t+4))$ $(t_4^\infty, \frac{7}{10} \times (t+3))\}$	(t_0^∞) : v_{10}, v_6, v_8
17	4	v_{10}	$(0, \frac{4}{10} \times (t+7))$	v_7, v_{13}

Table 1: The value of $W(v_i)$ which corresponds to the example in Fig. 2, ordered according to setup time of the algorithm in column no. 2.

2.2.1 Correctness of the Algorithm *Broadcast I*

In the following analysis we prove the correctness of Algorithm *Broadcast I*. Since the case where $|V(T)| \leq 2$ is correct and the proof is straight forward we assume that $|V(T)| > 2$. In addition, note that T_U is a subtree of T and hence it is connected.

We need additional notation:

$b^t(u, T)$ – the cost to broadcast from u a message received at time t. $b^0(u, T)$ is denoted by $b(u, T)$.

The following theorem deals with the cost to broadcast from v to $T_v^{(v,u)}$. The cost is shown to be dependent on the cost to broadcast from its sons. We remark that the function $f(T, v, t)$ used in the lemmas below stands for either the cost function $C(T, v, t)$ or $AC(T, v, t)$.

Theorem 2.1 *Let $u_1, u_2, \ldots, u_k$ be the neighbors of $v \in V(T)$. Suppose that $b(u_i, T_{u_i}^{(v,u_i)} \cup e_{u_i}^v) \geq b(u_{i+1}, T_{u_{i+1}}^{(v,u_{i+1})} \cup e_{u_{i+1}}^v)$ for $i = 1, 2, \ldots, k$ then:*

1. *For the cost defined in* 2.1.1*:*

$$b^t(v,T) = \max_{1 \leq i \leq k} \{\max\{b^{t+i}(u_i, T_{u_i}^{(v,u_i)}), w(e_v^{u_i}) \times (t+i)\}\}.$$

2. *For the cost defined in* 2.1.2*:*

$$b^t(v,T) = \frac{\sum_{i=1}^{k}(b^{t+i}(u_i, T_{u_i}^{(v,u_i)}) + w(e_v^{u_i}) \times (t+i))}{\sum_{i=1}^{k} n_{u_i} + 1}$$

where $n_{u_i} = |V(T_{u_i}^{(v,u_i)})|$.

Proof: The vertex v transmits the message from time $t+1$ until $t+k$ to all of its adjacent vertices u_1 to u_k. Since $b(u_i, T_{u_i}^{(v,u_i)} \cup e_{u_i}^v) \geq b(u_{i+1}, T_{u_{i+1}}^{(v,u_{i+1})} \cup e_{u_{i+1}}^v)$ for $i = 1, 2, \ldots, k$, and Observation I it follows that the value of $b^t(v,T)$ is minimum in the cases $1, 2$ of the theorem as required. ■

The next lemma proves that the value of the cost function assigned to the vertex $v \in V(T)$ in the algorithm *Broadcast I* equals the cost to broadcast from v to its sons.

Lemma 2.1 *Let $u \in L_{\bar{U}}$ be the vertex selected in step* 3.1*. Let v be the father of u in T_U and assume that u receives the message at time t. Then,*

$$f(T_u^{(v,u)}, u, t) = b^t(u, T_u^{(v,u)}).$$

Proof: This follows from the definition of $f(T, v, t)$ which is defined according to the requirements of theorem 2.1, and from the subtree $T_u^{(v,u)}$ on which the algorithm operates in steps 3.1 and 3.4. Namely, in step 3.1 we choose the father of u with the minimum value of the cost function f and update its value according to step 3.4. Therefore, conditions of Theorem 2.1.

The initial value of $W(u)$, which is $\{(t_0^\infty, 0)\}$ in the case where u is a leaf, matches the initial value zero of $f(T, v, t)$. ■

The following lemma shows that the cost function at time $t = 0$ is assigned non-decreasing values by the algorithm.

Lemma 2.2 *Let $f(T, u_i, t)$ be the cost functions for the pair $(t_0^z, f(T, u_i, t)) \in W(u_i)$ at $t \in t_0^z$, $z \geq 0$ and let v_i be the father of u_i in T_U, $i = 1, \ldots n-1$. If $u_1, u_2, \ldots u_{n-1}$ is the sequence of vertices selected in step* 3.1 *of the algorithm Broadcast I and u_n is the vertex that was remained after the end of step* 4*, then,*

$$f(T_{u_1}^{(v_1,u_1)}, u_1, 0) \leq f(T_{u_2}^{(v_2,u_2)}, u_2, 0) \leq \ldots \leq f(T, u_n, 0)$$

Proof: Let l, m be integers such that $1 \leq l < m \leq n$. It is sufficient to show that $f(T_{u_l}^{(v_l,u_l)}, u_l, 0) \leq f(T_{u_m}^{(v_m,u_m)}, u_m, 0)$. The vertex u_l was selected during the $l-$th iteration of step 3.1 and the vertex u_m was selected during the $m-$th iteration of step 3.1. If all adjacent vertices of u_l in $T_{u_l}^{(v_l,u_l)}$ and all adjacent vertices of u_m in $T_{u_m}^{(v_m,u_m)}$ were in $L_{\bar{U}}$ before u_l was selected then according to the way the vertices are selected in step 3.1 and by the cost function which is non-decreasing with time increase we have $f(T, u_l, 0) \leq f(T, u_m, 0)$. If the above adjacent vertices were not included in the same time in $L_{\bar{U}}$ then because u_l was selected before u_m and because the cost function which is non-decreasing with time increase we have $f(T, u_l, 0) \leq f(T, u_m, 0)$. ■

Lemma 2.3 *Let u and v be adjacent vertices in T, and assume that $b(u, T_u^{(v,u)}) \leq b(v, T_v^{(u,v)})$, then:*

$$b(u, T) \geq b(v, T)$$

Proof: The cost $b(u, T_u^{(v,u)})$ does not take into consideration the cost to broadcast from u to $T_v^{(u,v)}$. Since $b(v, T_v^{(u,v)}) \geq b(u, T_u^{(v,u)})$ and since the cost function is assigned non-decreasing values then $b(u, T) \geq b(v, T)$. ■

We prove now that a vertex of the *broadcast center* of T remains in the sub-tree T_U during the execution of the algorithm, namely,

Theorem 2.2 $Center(T) = BC(T)$.

In order to prove the theorem we need the following lemmas.

Lemma 2.4
In steps 3 and 4 of the algorithm, the broadcast center of T contains a vertex of T_U.

Proof: Let u be a vertex selected in step 3.1 and let v be the father of u in T_U. It is sufficient to show that $b(v, T) \leq b(u, T)$. Namely, the vertex with the lower broadcast value remains in T_U.

First, we show that $b(u, T_u^{(v,u)}) \leq b(v, T_v^{(u,v)})$. If $V(T_U) = \{u, v\}$ (only two vertices are in T_U), then according to lemma 2.1 the values of the cost function f for u and v are $b(u, T_u^{(u,v)})$ and $b(v, T_v^{(v,u)})$, respectively. Moreover, according to the vertices selection condition in step 3.1 of the algorithm we get $f(T, u, t) \leq f(T, v, t)$. Hence, $b(u, T_u^{(v,u)}) \leq b(v, T_v^{(u,v)})$.

Second we treat the case where $|V(T_U)| > 2$. Suppose that T_U has a leaf $x \notin \{v, u\}$ and let y be the father of x in T_U. By lemmas 2.1 and 2.2 and since u was selected in step 3.1 we have $b(x, T_x^{(y,x)}) \geq b(u, T_u^{(v,u)})$. Since $T_x^{(y,x)}$ is a subtree of $T_v^{(u,v)}$ (u is a leaf in T_U) and since the cost function assign non-decreasing values, we have $b(u, T_u^{(v,u)}) \leq b(v, T_v^{(u,v)})$. Therefore, by using lemma 2.3 we get $b(u, T) \geq b(v, T)$. ■

Corollary 2.1 *Let v be the vertex remained in $V(T_U)$ after step 3. Then, $v \in BC(T)$.*

Proof: Let $v \in V(T_U)$ be the vertex that was remained in $V(T_U)$ after step 3 of the algorithm ($|V(T_U)| = 1$). By lemma 2.4 this vertex is in $BC(T)$. ■

Lemma 2.5 *After step 4 of the algorithm, $Number(T)$ is the broadcast number of T.*

Proof: Let $v \in V(T_U)$ be the vertex that was remained in $V(T_U)$ after step 3 of the algorithm. By lemma 2.4 $v \in BC(T)$. Hence, by lemma 2.1 and lemma 2.2 $b(v,T)$ is the *broadcast number* of T. $Number(T)$ is assigned the value of $f(T,v,0)$ which is equal to $b^0(v,T)$ by lemma 2.1 (recall that $b(v,T)$ is denoted as $b^0(v,T)$). ■

The next lemma confirms that the value of $b(v,T)$ is minimal for v which is the last vertex that was remained after step 3.

Lemma 2.6 *Let v be the vertex that was obtained in step 3. Suppose $u \in L_{\bar{U}}$ is its son and $w \in V(T_u^{(v,u)}) \setminus \{u\}$. Then, $b(v,T) < b(w,T)$.*

Proof: Let x be an adjacent vertex to u in the path $(u - w)$. Obviously, $b(v, T_u^{(v,u)} \cup \{v\}) \geq b(u, T_u^{(v,u)}) \geq b(x, T_x^{(u,x)})$. By lemma 2.3 we have:

$$b(v,T) \leq b(u,T) \leq b(x,T).$$

If there is at least one sharp inequality in the above inequality then we are done. Otherwise, $b(v,T) = b(u,T) = b(x,T)$. If $b(v,T)$ is obtained in $T_v^{(u,v)}$, then, $b(u,T) > b(v,T)$ since u has to transmit the message to v and only at the next time unit it is transmitted to its adjacent vertices, which is a contradiction. The same is true for u and x. Hence, if $b(x,T)$ was obtained in $T_x^{(u,x)}$ then $b(x,T) > b(u,T)$ which is a contradiction.

If $b(v,T)$ is obtained in $T_u^{(v,u)}$ and $b(x,T)$ is obtained in $T_x^{(u,x)} \cup \{u\}$, then, by the same arguments we have $b(w,T) > b(x,T)$ as required. ■

Finally, we equate both notions of $Center(T)$ and the *Broadcast center* of T.

Proof of Theorem 2.2: The size of $BC(T)$ is at least 1 since there exists a vertex such that its broadcasting number is minimal.

First, we show that each $w \in Center(T)$ is in $BC(T)$. Let $v \in V(T_U)$ be the vertex which remained in $V(T_U)$ after step 3. Then, $v \in Center(T)$ (step 4) and $v \in BC(T)$ by lemma 2.4. Assume that $|Center(T)| > 1$, $w \in Center(T)$ and $w \neq v$. Then, $b(w,T) = b(v,T)$. By lemma 2.5 $Number(T) = b(v,T)$, therefore, $w \in BC(T)$.

Second, we show that for any $w \in V(T)$ and $w \notin Center(T)$ we have that $w \notin BC(T)$. Let $w \in V(T)$, $w \notin Center(T)$. We have the following two cases:

1. **Vertex w is adjacent to v:** Since w was not selected to be in $Center(T)$ in step 4 of the algorithm then, $b(w,T) > b(v,T)$.

2. **Vertex w is not adjacent to v:** $b(w,T) > b(v,T)$ according to lemma 2.6.

■

2.3 Complexity Analysis

In this section we show that the algorithm *Broadcast I* has time and space complexity of $O(n^2 \log n)$.

In step 1, we assign a value to $W(v)$ for each $v \in U$ where U is the set of leaves of T. Since $|U| \leq n$ this step requires at most $O(n)$ time.

In step 2 the leaves in T_U are located in $O(n)$ time. In addition, clearly $|L_{\bar{U}}| < n$. Therefore, this step is performed at most $O(n)$ time. In addition, for each $v \in L_{\bar{U}}$ we need the value of the cost function $f(T, v, t)$. This is done by sorting the set U in $O(n \log n)$ in the beginning of step 2. In this step the value of $f(T, v, t)$ gets only one maximum value for the time interval t_0^∞, therefore, the loop in step 2.1 is executed only once. Hence, step 2 requires $O(n \log n)$ time.

The loop in step 3 is executed at most $O(n)$ times since in each iteration one vertex is removed from T_U. The value of $f(T, v, 0)$ is kept in the list (step 3.2). Each time a vertex is assigned with this value it is added to list. Since the cost function $f(T, v, t)$ have non-decreasing values, then the first vertex in the list has a minimum cost. Therefore, step 3.1 requires $O(1)$ time. Clearly, step 3.2 requires $O(1)$ time.

Before examining step 3.4 we note that at most $O(1)$ vertices may be adjacent to $O(n)$ vertices in the tree. Therefore, the cost to sort the costs of the sons of a vertex is $O(n \log n)$ in $O(1)$ times. Since the time to broadcast in a tree is bounded by $n-1$ (see [4]) we may have up to $n-1$ different maximum values for $f(T, v, t)$ for different time intervals. Therefore, step 3.4 requires $O(n^2 \log n)$ time for $O(1)$ vertices. In all other cases it requires up to $O(n)$. Hence, the time complexity of step 3 is $O(n^2 \log n)$.

The time complexity of step 4 depends on the number of neighbors, say k, of the last vertex v which was remained in T_U after step 3. Then, k is at most $O(n)$. Since we need to calculate f for each adjacent vertex where their list of adjacent vertices were sorted during step 3, the complexity of step 4.1 is $O(n \log n)$. Hence, the complexity of step 4 is $O(n \log n)$. Therefore, the total complexity of the algorithm *Broadcast I* is $O(n^2 \log n)$.

Note that for the case where all weights are equal we get only single time interval (t_0^∞). Therefore, steps 2.1 and 3.4 of the algorithm are performed only once. In addition, we do not need to sort U, the set of leaves of T, prior to step 2 and we maintain a list of handled vertices for each $v \in V(T)$ as in [15]. Therefore, steps 2, 3 and 4 require $O(n)$ time and the time complexity of the algorithm is $O(n)$.

2.4 The Cost to Broadcast from Any Vertex

The cost to broadcast from any vertex is computed using the values of $W(v)$, $v \in V(T)$, which was computed in algorithm *Broadcast I*.

The following lemma enables us to compute the minimum cost to broadcast from any vertex by updating the cost on the path from the originator to a vertex in $BC(T)$.

Lemma 2.7 *Let $u \in V(T)$ be such that $u \notin BC(T)$ and let $v \in BC(T)$ be the vertex which has the shortest path to u. Let $x_1, x_2, \ldots, x_k$, $k \geq 2$, be the vertices in the path from v to u ($v = x_1$, $u = x_k$), where $(x_i, x_{i+1}) \in E(T)$. Then, the cost to broadcast from u is obtained by recomputing $W(x_i)$ for $i = 1 \ldots k$.*

Proof: Since T is a tree, where only one path exists between any two vertices, then, $x_1, x_2, \ldots, x_k$ are the only vertices that their cost functions are modified. Namely, the value of $f(T, w, t)$ where $w \in V(T), w \notin \{x_i\}_{i=1}^k$ is the same as the value while broadcasting from v. The cost to broadcast from x_k $(= u)$ is evaluated by the cost to broadcast to the sub-trees rooted by its adjacent vertices

(Theorem 2.1). This cost, which by lemma 2.1 is calculated using the cost function f of the adjacent vertices, is known for all the vertices adjacent to x_k but x_{k-1}. This is due to the procedure described above and to the order of the vertices handled by algorithm *Broadcast I* (from x_k to x_1) which computes the value of $f(T_{x_i}^{(x_{i+1},x_i)}, x_i, t)$ and not $f(T_{x_{i-1}}^{(x_i,x_{i-1})}, x_i, t)$.

The only vertex from $x_i \in \{x_j\}_{j=1}^{k}$ that the value of its $f(T_{x_{i-1}}^{(x_i,x_{i-1})}, x_i, t)$ can be computed immediately, is x_2, since it is computed using the cost of its adjacent vertices including x_1 $(= v)$ which are not modified. The cost of the rest of the vertices $x_i, i = 3, \ldots, k$ is computed based upon the cost of x_{i-1}. Finally, the cost to broadcast from x_k $(= u)$ is the value of $f(T, x_k, 0)$ as in step 4 of the algorithm *Broadcast I.* ■

2.5 Broadcasting Multiple Messages

In order to broadcast $m(\geq 1)$ messages from any vertex $v \in V(T)$ we have to find $BC(T)$ and $b(T)$.

Initially, we deal with the case $m = 1$. Let $v \in V(T)$ with $u_1, u_2, \ldots, u_k$ as its neighbors. They are labeled in an non-descending order according to their cost value (the cost to broadcast to their weighted sub-tree related to the cost to transmit from v to u_j, $j = 1, \ldots, k$). The broadcasting pattern is such that v receives the message from u_l, $1 \leq l \leq k$, at time t and transmits it to its adjacent vertices u_j, $j = 1, \ldots, k$, $j \neq l$, in the following order: To u_1 at time $t + 1$, to u_2 at time $t + 2$, until at time $t + k - 1$ the message is transmitted to u_k. Observe that if v is the originator then the restriction upon u_l is removed, namely, v transmits the message to all its neighbors.

For broadcasting $m > 1$ messages the originator v_0 transmits the first message to its adjacent vertices, as in the previous case, then it waits for a constant time step, t_{v_0}, and transmits the next message with the same pattern as before. The constant t_{v_0} is the smallest positive integer such that $t_{v_0} \geq \Delta - d(v_0)$, where Δ is the maximal degree in T and $d(v_0)$ is the degree of v_0.

Assume that v_0 is the originator and let $v \in V(T), v \neq v_0$ and suppose v receives a message in time t. Then v transmits the message to its neighbors (excluding the one from which v received the message) in time $t + 1$ to $t + d(v) - 1$ as in the case $m = 1$.

3 Broadcasting in a vertex-weighted tree

In a vertex-weighted tree T the weights $w(v)$ are assigned to all $v \in V(T)$ or only to all $l \in L$, where L is a set of leaves in T. Broadcasting is performed according to ϵ-broadcast or δ-broadcast that are defined below. The weight of a vertex in the ϵ-broadcast is a threshold for the vertex to receive the message. Namely, only vertices with weights greater equal to ϵ receive the message. The above is often called a multicast[11]. This is different from the δ-broadcast where the weight of a vertex is considered as the time the vertex needs to process the message prior to transmitting it to its adjacent vertices. Namely, a vertex receives a message, then process it for a given time (its weight) and send it to its adjacent vertices only after δ time units. Notice that we can combine the above broadcasting algorithms with the edge weighted algorithm. This enable us to prioritize vertices in either the ϵ-broadcast or in the δ-broadcast case.

3.1 Formal Definition of the Problem

The problem is how to broadcast $m(\geq 1)$ messages from any vertex $v \in V(G)$ in a given graph G with weights $w(v)$ that were assigned to all vertices $v \in S \subseteq V(G)$. This problem uses the following two definitions:

ϵ-broadcast means broadcasting m messages with a minimum *cost* to all vertices in $S \triangleq \{v \in V(G) \mid w(v) \geq \epsilon\}$. The *cost* was defined in section 2.1 where $w(v)$ replaces $w(e)$ and $t(v)$ replaces $t(e)$, where $t(v)$ is the time a vertex v receives a message. The weight of a vertex in $V(G) \setminus S$ is assumed to be zero. Namely, if a transmitted message which arrives from a vertex x to a vertex y passed through a vertex z where $w(z) < \epsilon$ we assume that $w(z) = 0$ in the cost calculations. Fig. 3 demonstrates the definition where x is the originator and $\epsilon = \frac{2}{10}$ (the weights on the vertices are shown below their symbols).

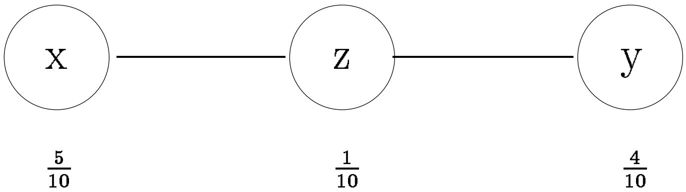


Figure 3: ϵ-broadcast in a vertex-weighted tree

$\delta(v)$-broadcast means broadcasting m messages in a minimum *cost* to all vertices $v \in V(G)$ where for each vertex $\{v_i \mid 1 \leq i \leq |V|\}$ the message is delayed for $\delta(v_i) = w(v_i)$ time units, and $w(v_i)$ are positive integer values. If v_i receives a message at time t then it is transmitted at time $t + w(v_i) + 1$ to its first neighbor, at time $t + w(v_i) + 2$ to its second neighbor, and so on until at time $t + w(v_i) + d(v_i)$ ($d(v_i)$ is the degree of v_i in G) it transmits the message to its last neighbor. The *cost* is defined to be the time needed to complete the broadcasting.

Again, the *broadcast center* of G, $BC(G)$, is defined as before to be the set of all vertices having minimal broadcasting number.

3.2 Locating the Broadcast Center for Vertex-Weighted Tree

3.2.1 ϵ-broadcast - *Broadcast II*

This algorithm is based upon Algorithm *Broadcast I*. It differs from it because we consider the weight of a vertex instead of an edge. Furthermore, in case $w(v) < \epsilon$ we assume $W(v)$ to be zero. We omit the presentation of algorithm *Broadcast II* which is similar to algorithm *Broadcast I*.

The value of the cost function, $C_1(T, v, t)$, is computed according to the cost definition, which is the broadcasting time, as follows: Let v be any vertex in V with $u_1, u_2, \ldots, u_k$ as its neighbors. Then, we compute the cost function according to the following:

1. For the cost measurement as in 2.1.1 the cost function of v, $C_1(T, v, t)$, is:

$$\max\left\{\max_{1\leq i\leq k}\left\{C_1(T_{u_i}^{(v,u_i)}, u_i, t+i)\right\}, \omega_v \times t\right\},$$

   where

$$\omega_v = \left\{\begin{array}{ll} w(v), & w(v) \geq \epsilon \\ 0, & otherwise \end{array}\right\}$$

   The neighbors of v are ordered such that $C_1(T_{u_i}^{(v,u_i)} \cup e_{u_i}^v, u_i, t) \geq C_1(T_{u_{i+1}}^{(v,u_{i+1})} \cup e_{u_{i+1}}^v, u_{i+1}, t)$, $1 \leq i \leq k-1$. For a leaf v in the tree T, $C_1(T, v, t) = \omega_v$.

2. For the cost measurement as in 2.1.2, we keep the value of the cost function of each vertex in the tree, and the number of vertices for which this function was calculated. Then, the cost function of v, $AC_1(T, v, t)$, is:
$$\frac{\sum_{i=1}^{k}\left(AC_1(T_{u_i}^{(v,u_i)}, u_i, t+i) + \omega_v \times t\right)}{\sum_{i=1}^{k} n_{u_i} + 1},$$
where ω_v is defined as in the above. The neighbors of v are ordered as in 1 and n_{u_i} is the number of vertices in $T_{u_i}^{(v,u_i)}$. For a leaf v in the tree T, $n_v = 1$ and $AC_1(T, v, t) = \omega_v$.

Notice that as in Algorithm *Broadcast I* we assign the functions AC_1 or C_1 instead of f according to the required cost measurements.

Remark:

In the case of equal weights, say, $w(v) = 1 \;\forall\; w(v),\; v \in V(T)$ we get only one time interval (t_0^∞). Therefore, steps 2.1 and 3.4 of the algorithm are performed only once and algorithm *Broadcast II* coincides with the algorithm presented in [15]. ■

Correctness and Complexity of *Broadcast II*

The proof of the correctness of this algorithm is similar to the proof of *Algorithm I.* The only difference is that the weights were assigned to vertices instead of edges. Note that the cost functions consider the weight of a vertex v assign the value zero whenever $w(v) < \epsilon$.

The complexity of *Broadcast II* is $O(n^2 \log n)$ as of algorithm *Broadcast I.* Again, when all weights are equal we have time complexity $O(n)$.

3.2.2 δ-broadcast - *Broadcast III*

Broadcast III is based upon *Broadcast II* when $\epsilon = \infty$. The only difference is in the definition of the cost function. We get only one time interval (t_0^∞). Therefore, steps 2.1 and 3.4 of the algorithms are executed only once.

In the case, $w(v) = 0,\; \forall\; v \in V(T)$ the message is not delayed. Moreover, we get only one time interval (t_0^∞) and this algorithm coincides with the algorithm presented in [15].

The value of the cost function, $C_2(T, v, t)$, is computed according to the cost definition, which is the broadcasting time, as follows: Let $v \in V(T)$ with $u_1, u_2, \ldots, u_k$ as its neighbors. Let the neighbors of v be ordered such that $C_2(T_{u_i}^{(v,u_i)} \cup e_{u_i}^v, u_i, t) \geq C_2(T_{u_{i+1}}^{(v,u_{i+1})}, u_{i+1} \cup e_{u_{i+1}}^v, t)$, $1 \leq i \leq k-1$. Then,
$$C_2(T, v, t) = w(v) + \max_{\{1 \leq i \leq k\}} \{C_2(T, u_i, t) + i\}.$$
Note that if v is a leaf of T then the value of the cost function is $w(v)$.

Correctness and Complexity of the Algorithm

The proof for correctness is similar to the proof for *Broadcast I* and *Broadcast II.*

In the case of equal weights the complexity of this algorithm is the same as algorithm *Broadcast I*, since we get only one time interval (t_0^∞). Therefore, the steps 2.1 and 3.4 of the algorithms are executed only once. In addition, we do not need to sort U, the set of leaves of T, prior to step 2 and we can maintain a list of handled vertices for each $v \in V(T)$ as in [15]. Therefore, the time complexity of the algorithm is $O(n)$.

3.3 The Cost to Perform ϵ-broadcasting or δ-broadcasting from Any Vertex

The algorithms for ϵ-broadcast and δ-broadcast from any vertex v are based upon the *broadcast number* of v which is computed using the *broadcast center* of T. This is done similarly to the edge-weighted tree (section 2.4). The *broadcast number* of v, $b_\epsilon(v)$ or $b_\delta(v)$ is the cost to perform ϵ-broadcast for a given ϵ from v or δ-broadcast from v, respectively (section 3.1). It is computed, as in the edge-weight broadcasting, using $BC_\epsilon(T)$ or $BC_\delta(T)$ where $BC_\epsilon(T)$ and $BC_\delta(T)$ denote the $BC(T)$ for ϵ-broadcast or δ-broadcast, respectively.

3.4 Broadcasting Multiple Messages

As in the edge-weighted tree, we need to find $BC(T)$ and $b(v)$ before broadcasting $m(\geq 1)$ messages from any vertex $v \in V$ in a given tree T. For the ϵ-broadcast the algorithm to broadcast $m(\geq 1)$ messages is identical to the edge-weighted tree.

Broadcasting $m \geq 1$ messages for the δ-broadcast is done as follows: First we deal with the case where $m = 1$. Let v_0 be the originator. Let v be any vertex in V with $u_1, u_2, \ldots, u_k$ as its neighbors that are labeled in a non-descending order according to their cost function values. The broadcast pattern is such that v receives the message from u_l, $1 \leq l \leq k$, at time t, transmits it to u_1 at time $t + w(v) + 1$, to u_2 at time $t + w(v) + 2$ and so on except for u_l, until at time $t + w(v) + k - 1$ the message is transmitted to u_k. If $v = v_0$ then the restriction upon u_l is removed.

For broadcasting $m > 1$ messages the originator v_0 transmits the first message to its adjacent vertices, as in the previous case, then waits for constant time steps, t_{v_0}, and transmits the next message in the same pattern. We have that

$$t_{v_0} = \max_{v \in V(T)} \{w(v) + d(v)\} - d(v_0) - w(v_0) + 1,$$

where $d(v)$ is the degree of v.

All other vertices, $v \neq v_0$, transmit each message received in time t in the same pattern as in the case $m = 1$ from time $t + w(v) + 1$ to $t + w(v) + k - 1$.

Acknowledgment

The authors like to thank the referee for his suggestions and remarks.

References

[1] A. Averbuch, Y. Roditty, B. Shoham, "On Scheduling Tasks of Multi-Join Queries in a Multiprocessor", *Concurrency: Practice and Experience, To appear.*

[2] A. Averbuch, Y. Roditty, B. Shoham, "On Broadcasting Messages in Communication Networks", *Submitted.*

[3] A. Averbuch, Y. Roditty, B. Shoham, "On Time - Relaxed Broadcasting in Communication Networks", *Submitted.*

[4] R. Bar-Yehuda, O. Goldrich and A. Itai. "On the Time-Complexity of Broadcast in Radio Networks: An Exponential Gap Between Determinism and Randomization", *J. of Computer and System Sciences* **45** (1992), 104-126.

[5] P. Chinn ,S Hedetniemi and S. Mitchell, "Multiple message broadcasting in complete graphs." *Proceedings of the 10th SE Conference on Combinatorics, Graph Theory and Computing. Utilitas Math. Winnipeg (1979), 251-260* .

[6] E. J. Cockayne and A. Thomason, "Optimal multi-message broadcasting in complete graphs." *Proceedings of the 11th SE Conference on Combinatorics, Graph Theory and Computing. Utilitas Math. Winnipeg (1980), 181-199.*

[7] M. R. Garey, D. S. Johanson, "A Guide to the Theory of NP-Completeness", *Freeman, 1979.*

[8] S. M. Hedetniemi, S.T. Hedetniemi and A.L. Liestman. "A Survey of Gossiping and Broadcasting in Communication Networks", *Networks* **18** (1988), 319-349.

[9] P. Hell, K. Seyffarth. "Broadcasting in Planar Graphs", *Preprint.*

[10] Oh-Heum Kwon and Kyung-Yong Chwa, "Multiple messages broadcasting in communication networks." *Networks 26(1995), 253-261.*

[11] L. Liang, S. T. Chanson and G. W. Neufeld. "Process Groups and Group Communications: Classifications and Requirements", Computer, February 1990, pp. 56-66.

[12] Y. Roditty, B. Shoham, "On Broadcasting Multiple Messages in a $d-$dimensional Grid." *Discrete Applied Mathematics, 75(1997), 277-284.*

[13] A. Shastri, "Time-relaxed broadcasting in communication networks." *Discrete Applied Mathematics, 83(1998), 263-278.*

[14] A. Shastri, "Broadcasting in general networks I: Trees." *Proceedings of Cocoon (1995), Xian China.*

[15] P. J. Slater, E.J. Cockayne and S.T. Hedetniemi, "Information Dissemination in Trees", *SIAM J. Comput.* **10** (1981) No. 4, 692-701.

[16] F. L. Van Scoy and J. A. Brooks, "Broadcasting multiple messages in a grid." *Discrete Applied Mathematics 53(1994), 321-336.*