Nonadaptive Broadcasting in Trees

Hovhannes A. Harutyunyan
Department of Computer Science and Software Engineering, Concordia University, Montreal, Quebec, Canada H3G 1M8

Arthur L. Liestman
School of Computing Science, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6

Kazuhisa Makino
Department of Mathematical Informatics, Graduate School of Information and Technology, University of Tokyo, Tokyo 113-8656, Japan

Thomas C. Shermer
School of Computing Science, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6

We study nonadaptive broadcasting in trees, a process of sending a message from one vertex in a tree to all other vertices. In the nonadaptive model, each vertex has a specified, ordered list of its neighbors. After receiving a broadcast message, a vertex sends the message to its neighbors, one after another, in the order specified by the list. The broadcast is completed when all vertices have received the message. We obtain lower and upper bounds on the minimum time required to complete a nonadaptive broadcast in a tree and improved upper bounds for general graphs. We give a polynomial time algorithm for determining the minimum nonadaptive broadcast time of any given tree. We also show how to construct the largest possible trees having a given nonadaptive broadcast time. **© 2010 Wiley Periodicals, Inc.**
NETWORKS, Vol. 57(2), 157–168 2011

Keywords: broadcasting; path; tree; universal list; nonadaptive

1. INTRODUCTION

Broadcasting is a process of disseminating information from a single vertex (the originator) of a connected graph to all other vertices. Vertices that have the message inform their neighbors and the process continues until all vertices have the message at which point we say that the broadcast is complete. Many variations of broadcasting have been studied.

Received August 2009; accepted March 2010
Correspondence to: A. L. Liestman; e-mail: art@cs.sfu.ca
Contract grant sponsor: Natural Sciences and Engineering Research Council of Canada
DOI 10.1002/net.20396
Published online 12 November 2010 in Wiley Online Library (wileyonlinelibrary.com).

It is common to assume that each vertex that has the message can send it to at most one of its neighbors in a given time unit. We assume that

1. A vertex can receive a message from several of its neighbors simultaneously.
2. When a vertex first receives the message, it (starting at the next time unit) sends the message to each of its neighbors, one per time unit, with no repetitions, in some order. We call this order the broadcast list for this vertex.
3. The originator is assumed to receive the message at time 0.

In the classical broadcast model, each vertex chooses its broadcast list based solely on which vertex is the originator of the broadcast. That is, a vertex may choose different broadcast lists for different originators.

In broadcasting with universal lists, each vertex has a fixed (universal) list which is used in every broadcast, regardless of which vertex originated the broadcast.

In adaptive broadcasting with universal lists, this restriction is relaxed slightly. Here, each vertex knows which vertex it received the message from and such a vertex is omitted from the otherwise fixed broadcast list.

In nonadaptive broadcasting with universal lists, the fixed broadcast list is always used and, thus, each nonoriginating vertex will send the message back to the vertex from which it was received.

For simplicity, we shorten the names of these last two models to adaptive broadcasting and nonadaptive broadcasting.

In general, a broadcast scheme is a collection of broadcast lists for every vertex. There is one list per originator per vertex

in the classical model but only one list per vertex in either adaptive or nonadaptive broadcasting.

Informally, the broadcast time of a broadcast scheme is the latest time (over all originators) at which the broadcast defined by the scheme is complete. The broadcast time of a graph is the minimum broadcast time of any broadcast scheme for that graph.

Several articles have considered these models for broadcasting. Slater et al. [9] indirectly observed that the broadcast time for trees in the adaptive model is the same as in the classical model. Rosenthal and Scheuermann [8] introduced universal lists and gave an algorithm to construct a broadcast scheme with minimum broadcast time for trees in the adaptive model. Diks and Pelc [1] distinguished between the adaptive and nonadaptive models and determined the adaptive and nonadaptive broadcast times for cycles and two-dimensional grids. They also gave upper bounds on the adaptive and nonadaptive broadcast time for two-dimensional tori and complete graphs. They also gave upper and lower bounds on the nonadaptive broadcast time for trees. Harutyunyan and Taslakian [3] improved the upper bounds on nonadaptive broadcast time for two-dimensional tori. Kim and Chwa [7] constructed nonadaptive broadcast schemes for paths and two-dimensional grids and gave upper bounds on nonadaptive broadcast time for complete k-ary trees, complete graphs, and hypercubes.

In this article, we study nonadaptive broadcasting in trees. We give improved bounds on the nonadaptive broadcast time for trees, present a polynomial time algorithm for determining the nonadaptive broadcast time for any given tree, and show how to construct the largest possible trees having a given nonadaptive broadcast time.

There is an extensive literature about other variations of broadcasting (cf. [2, 4–6]).

2. DEFINITIONS AND PRELIMINARY RESULTS

Let $G = (V, E)$ be a connected graph on n vertices, let σ denote a broadcast scheme, and let s denote an originator. Following the notation of Diks and Pelc [1], let M be one of the models: classical (cl), adaptive (a), or nonadaptive (na). We define $B_{\mathrm{M}}(G, s, \sigma)$ to be the time required by the scheme σ to broadcast from s in G. $B_{\mathrm{M}}(G, \sigma)$ is the maximum of $B_{\mathrm{M}}(G, s, \sigma)$ over all originators s and $B_{\mathrm{M}}(G)$ is the minimum of $B_{\mathrm{M}}(G, \sigma)$ over all schemes σ for G. The value $B_{\mathrm{M}}(G)$ is called the broadcast time of the graph for model M.

We note that $B_{\mathrm{cl}}(K_n) = \lceil \log n \rceil$, and that if G is a connected graph on n vertices, then $\lceil \log n \rceil \leq B_{\mathrm{cl}}(G) \leq n-1$.

Consider a graph $G = (V, E)$ and assume that each vertex u has a single list L_u of (some of) its neighbors that comprise a scheme σ for broadcasting with universal lists. Given an originator $s \in V$ the nonadaptive broadcast from s proceeds as follows: at time 0, the originator s receives the message. For any vertex v which first receives the message at time t, vertex v sends the message to its neighbors in the order that they appear in list L_v at times $t+1, t+2, \ldots$. An adaptive broadcast proceeds in the same way except that every neighbor that has already sent the message to v is skipped. In either case, the broadcast is complete when every vertex has received the message. Note that the vertices may continue sending messages after the broadcast is complete.

The time at which the last vertex receives the message is denoted $B_{\mathrm{M}}(G, s, \sigma)$. $B_{\mathrm{M}}(G, \sigma)$ is the maximum of $B_{\mathrm{M}}(G, s, \sigma)$ over all originators $s \in V$ and $B_{\mathrm{M}}(G)$ is the minimum of $B_{\mathrm{M}}(G, \sigma)$ over all schemes σ for G. The value $B_{\mathrm{M}}(G)$ is called the broadcast time of the graph for model M.

In contrast to our definition for $B_{\mathrm{M}}(G, \sigma)$ which uses maximum, we define $b_{\mathrm{M}}(G, \sigma)$ to be the minimum of $B_{\mathrm{M}}(G, s, \sigma)$ over all originators $s \in V$. Similarly, we define $b_{\mathrm{M}}(G)$ to be the minimum of $b_{\mathrm{M}}(G, \sigma)$ over all schemes σ for G. The vertices s with $B_{\mathrm{M}}(G, s, \sigma) = b_{\mathrm{M}}(G)$ for some σ are called the broadcast center of the graph for model M [9].

We use $\mathrm{diam}(G)$ to denote the diameter of the graph G.

We note the following results shown by Diks and Pelc [1]:

Observation 2.1. *$B_{\mathrm{cl}}(G) \leq B_{\mathrm{a}}(G) \leq B_{\mathrm{na}}(G)$ for any G.*

From known bounds on $B_{\mathrm{cl}}(G)$, we get the following observations.

Observation 2.2. *$\lceil \log n \rceil \leq B_{\mathrm{a}}(G) \leq B_{\mathrm{na}}(G)$ for any graph G on n vertices.*

Observation 2.3. *$\mathrm{diam}(G) \leq B_{\mathrm{a}}(G) \leq B_{\mathrm{na}}(G)$ for any graph G.*

Theorem 2.4 (Theorem 2.1 of Ref. [1])**.** *$B_{\mathrm{cl}}(T) < B_{\mathrm{na}}(T) \leq B_{\mathrm{cl}}(T) + \lfloor \frac{B_{\mathrm{cl}}(T)}{2} \rfloor$ for any tree T with at least three vertices.*

In this article, we restrict our attention to the nonadaptive model, except when noted. This being the case, we will generally omit the subscript *na* unless needed. In Section 3, we give bounds on $B_{\mathrm{na}}(T)$ where T is an arbitrary tree. In Section 4, we present a polynomial time algorithm for determining $B_{\mathrm{na}}(T)$ given a tree T. In Section 5, we show how to construct large trees for given nonadaptive broadcast time. Finally, in Section 6 we give an improved upper bound for $B_{\mathrm{na}}(G)$, where G is an arbitrary graph.

3. TREES

Consider any nonadaptive broadcast scheme σ for an arbitrary tree T. Let $b_\sigma(i, j)$ denote the time required for a message originating at v_i to arrive at v_j under the scheme σ for T. Let $l_i(j)$ denote the position of the vertex v_j in L_{v_i}, that is, if v_i receives a message at time t, then v_i calls v_j at time $t + l_i(j)$. When we use unsubscripted vertices u and v, we will use the notations $b_\sigma(u, v)$ and $l_u(v)$. Note that $1 \leq l_i(j) \leq \deg(v_i)$.

We can improve on Observation 2.3 for trees as shown by the following lemma.

Lemma 3.1. *$\lceil \frac{3\,\mathrm{diam}(T)-1}{2} \rceil \leq B_{\mathrm{na}}(T)$ for any tree T.*

Proof. Let $d = \mathrm{diam}(T)$. Without loss of generality, we label the vertices of a diameter path $v_0, v_1, \ldots, v_d$. Consider $b_\sigma(0,d)$ and $b_\sigma(d,0)$. $b_\sigma(0,d) = l_0(1)+l_1(2)+\cdots+l_{d-1}(d)$ and $b_\sigma(d,0) = l_d(d-1) + l_{d-1}(d-2) + \cdots + l_1(0)$. As $l_i(i+1) + l_i(i-1) \geq 3$ for $1 \leq i \leq d-1$ and both $l_0(1)$ and $l_d(d-1)$ are at least one, we get $b_\sigma(0,d) + b_\sigma(d,0) \geq 3(d-1)+2 = 3d-1$. As $B_{\mathrm{na}}(T) \geq \max\{b_\sigma(0,d), b_\sigma(d,0)\}$, $B_{\mathrm{na}}(T) \geq \lceil \frac{3d-1}{2} \rceil$. ∎

Diks and Pelc [1] showed that for any tree T with at least three vertices, $B_{\mathrm{cl}}(T) < B_{\mathrm{na}}(T)$. Lemma 3.1 gives an improved lower bound on $B_{\mathrm{na}}(T)$ for many trees. For example, let T_k denote the binomial tree of height k. T_k has classical broadcast time $2k-1$ and diameter $2k-1$. The lower bound of Diks and Pelc gives $B_{\mathrm{na}}(T_k) \geq 2k$ for $k \geq 2$ while Lemma 3.1 gives $B_{\mathrm{na}}(T_k) \geq 3k-2$, improving the lower bound on $B_{\mathrm{na}}(T_k)$ for all $k \geq 3$.

For the next result, we will make use of some facts about classical broadcast schemes on rooted trees. Let T be a tree with root r. Let v be an arbitrary vertex in T and let T_v be the subtree rooted at v. Let $v_1, v_2, \ldots v_k$ be the children of v. Furthermore, let T_i denote the subtree rooted at v_i and t_i denote the classical broadcast time from originator v_i in T_i. We assume that the children of v are labelled such that $t_1 \geq t_2 \geq \cdots \geq t_k$. In our notation, $L_v = v_1, v_2, \ldots, v_k$. There is an optimal classical broadcast scheme S for the root r in T such that every vertex v calls its children in the order in which they are labelled (this follows from Ref. [9]). We call such a scheme well ordered.

Suppose that we have a well-ordered scheme S_r for originator r, and now consider broadcasting from another vertex x. As T denotes the tree with root r, let T^* denote the tree with root x. For any vertex v not on the path from x to r, v has the same children $v_1, v_2, \ldots, v_k$ in both trees. Let the path from x to r be $x = w_0, w_1, \ldots, w_p = r$. Vertex $x = w_0$ has one child (w_1) in T^* that is not a child in T. Vertex $r = w_p$ has one child (w_{p-1}) in T that is not a child in T^*. Any other path vertex w_i has a child (w_{i-1}) in T but not in T^* and a child (w_{i+1}) in T^* but not in T. Otherwise, any child of a path vertex is present in both trees T and T^*.

Let v be an arbitrary vertex of the tree and let L_v and L_v^* denote the ordered lists for v in S_r and S_x, respectively. We say that v calls its children in the same order in S_r and S_x if the common children appear in the same order in both L_v and L_v^*.

Let S_r and S_x be broadcast schemes for tree T with originators r and x, respectively. We will say that S_r and S_x are consistent if every vertex v calls its children in the same order in both S_r and S_x.

We observe that for any originator x in T, there is an optimal classical broadcast scheme consistent with S_r. This again follows from Ref. [9].

We now obtain an upper bound on $B_{\mathrm{na}}(T)$ [and hence on $B_{\mathrm{a}}(T)$] based on the classical broadcast time $B_{\mathrm{cl}}(T)$ and the diameter of T.

Theorem 3.2. $B_{\mathrm{na}}(T) \leq B_{\mathrm{cl}}(T) + \lceil \frac{\mathrm{diam}(T)}{2} \rceil$.

Proof. Let u be a vertex in the center of T and consider T to be rooted at u. Note that for any vertex v, $d(u,v) \leq \lceil \frac{\mathrm{diam}(T)}{2} \rceil$.

Consider a well-ordered classical broadcast scheme S for originator u. The scheme can be considered to consist of a list L_v^S of neighbors for each vertex v. From these lists, we construct a nonadaptive broadcast scheme σ for T, that is, we construct a list L_v for each vertex v as follows: Let $L_u = L_u^S$. Let v be a vertex other than u. In the tree T rooted at u, v's neighbors consist of a parent w and children $v_1, v_2, \ldots, v_k$ which appear in that order in L_v^S. To form the list L_v, simply prepend w to the list L_v^S. In other words, each vertex first calls its parent and then proceeds to call its children in an order consistent with S. The resulting nonadaptive scheme is called $\sigma(S,u)$.

For any three vertices w, a, and b with a on the path from w to b, let $v_{\mathrm{cl}}^w(a,b)$ denote the time taken to inform vertex b once a has been informed in an optimal classical broadcast scheme for originator w that is consistent with S.

We now consider the $b_{\mathrm{na}}^\sigma(x,y)$, the time required for a message originating at x to arrive at y under our nonadaptive scheme $\sigma(S,u)$. Let u' be the vertex after u on the path from u to y.

$$
\begin{aligned}
b_{\mathrm{na}}^\sigma(x,y) &\leq b_{\mathrm{na}}^\sigma(x,u) + b_{\mathrm{na}}^\sigma(u,y) \\
&\leq b_{\mathrm{na}}^\sigma(x,u) + b_{\mathrm{na}}^\sigma(u,u') \\
&\quad + b_{\mathrm{na}}^\sigma(u',y) \text{ by the triangle inequality} \\
&= d(x,u) + v_{\mathrm{cl}}^u(u,u') + \left(v_{\mathrm{cl}}^u(u',y) + d(u',y)\right) \\
&\leq v_{\mathrm{cl}}^x(x,u) + \left(v_{\mathrm{cl}}^x(u,u') + 1\right) \\
&\quad + \left(v_{\mathrm{cl}}^x(u',y) + \left(\left\lceil \frac{\mathrm{diam}(T)}{2} \right\rceil - 1\right)\right) \\
&= v_{\mathrm{cl}}^x(x,y) + 1 + \left\lceil \frac{\mathrm{diam}(T)}{2} \right\rceil - 1 \\
&\leq B_{\mathrm{cl}}(T) + \left\lceil \frac{\mathrm{diam}(T)}{2} \right\rceil.
\end{aligned}
$$

∎

The following result uses $b_{\mathrm{cl}}(T)$, the minimum broadcast time from any originator, in contrast to the previous result which uses $B_{\mathrm{cl}}(T)$, the maximum broadcast time from any originator.

Theorem 3.3. $B_{\mathrm{na}}(T) \leq b_{\mathrm{cl}}(T) + \mathrm{diam}(T)$.

Proof. We choose a vertex r in the classical broadcast center of T as the root. Consider a well-ordered classical broadcast scheme S for originator r consisting of a list L_v for each vertex v. As in the previous proof, we construct a nonadaptive broadcast scheme σ for T as follows: let $L_r = L_r^S$. To form the list L_v of any vertex v other than r, simply prepend the parent of v to the list L_v^S.

Consider any two vertices u, v in T and let a be their least common ancestor.

$$\begin{aligned} b^{\sigma}_{\mathrm{na}}(u,v) &= b^{\sigma}_{\mathrm{na}}(u,a) + b^{\sigma}_{\mathrm{na}}(a,v) \\ &= d(u,a) + b^{\sigma}_{\mathrm{na}}(a,v) \\ &= d(u,a) + d(a,v) + b_S(a,v) \end{aligned}$$

where $b_S(a,v)$ is the time required for the message to travel from a to v under scheme S.

As $b_S(a,v) \le b_S(r,v) = b_{\mathrm{cl}}(T)$,

$$\begin{aligned} b^{\sigma}_{\mathrm{na}}(u,v) &\le d(u,a) + d(a,v) + b_{\mathrm{cl}}(T) \\ &= \mathrm{diam}(T) + b_{\mathrm{cl}}(T), \end{aligned}$$

establishing the upper bound. ∎

Each of the three upper bounds—those of Diks and Pelc, Theorem 3.2, and Theorem 3.3—is the best for some trees.

Consider the family of paths P_n with n vertices where n is even. $B_{\mathrm{cl}}(P_n) = \mathrm{diam}(P_n) = n-1$ and $b_{\mathrm{cl}}(P_n) = \frac{n}{2}$. The upper bound of Diks and Pelc gives $B_{\mathrm{na}}(P_n) \le \lfloor \frac{3(n-1)}{2} \rfloor = \frac{3n}{2} - 2$, Theorem 3.2 gives $B_{\mathrm{na}}(P_n) \le (n-1) + \lceil \frac{(n-1)}{2} \rceil = \frac{3n}{2} - 1$, and Theorem 3.3 gives $B_{\mathrm{na}}(P_n) \le \frac{n}{2} + (n-1) = \frac{3n}{2} - 1$. In this case, the upper bound of Diks and Pelc is the best.

Consider the complete bipartite graph $K_{1,n-1}$ on n vertices. $B_{\mathrm{cl}}(K_{1,n-1}) = b_{\mathrm{cl}}(K_{1,n-1}) = n-1$ and $\mathrm{diam}(K_{1,n-1}) = 2$. The upper bound of Diks and Pelc gives $B_{\mathrm{na}}(K_{1,n-1}) \le (n-1) + \lfloor \frac{n-1}{2} \rfloor$, Theorem 3.2 gives $B_{\mathrm{na}}(K_{1,n-1}) \le n$, and Theorem 3.3 gives $B_{\mathrm{na}}(K_{1,n-1}) \le n+1$. In this case, Theorem 3.2 gives the best upper bound for all $n \ge 5$.

Consider the family of trees T_n on n vertices (for even n) constructed as follows: begin with a path of $\frac{n}{2} + 1$ vertices. Attach $\frac{n}{2} - 1$ pendant vertices to one of its endpoints. $B_{\mathrm{cl}}(T_n) = n - 1$, $\mathrm{diam}(T_n) = \frac{n}{2} + 2$ and $b_{\mathrm{cl}}(T_n) = \frac{n}{2}$. For these graphs, the upper bound of Diks and Pelc gives $B_{\mathrm{na}}(T_n) \le n + \frac{n}{2} - 1$, Theorem 3.2 gives $B_{\mathrm{na}}(T_n) \le n + \frac{n}{4}$, and Theorem 3.3 gives $B_{\mathrm{na}}(T_n) \le n+2$. In this case, Theorem 3.3 gives the best upper bound for all $n \ge 8$.

Combining our new bounds with the bounds from Diks and Pelc [1], we have the following result:

Theorem 3.4. *For any tree T on at least three vertices,*

$$\begin{aligned} \max\left\{ B_{\mathrm{cl}}(T) + 1, \left\lceil \frac{3\,\mathrm{diam}(T) - 1}{2} \right\rceil \right\} \\ \le B_{\mathrm{na}}(T) \le \min\left\{ \begin{array}{l} \left\lfloor \frac{3B_{\mathrm{cl}}(T)}{2} \right\rfloor, \\ B_{\mathrm{cl}}(T) + \left\lceil \frac{\mathrm{diam}(T)}{2} \right\rceil, \\ b_{\mathrm{cl}}(T) + \mathrm{diam}(T) \end{array} \right\}. \end{aligned}$$

4. COMPUTING THE NONADAPTIVE BROADCAST TIME OF A TREE

To determine the nonadaptive broadcast time of a tree, we first develop an algorithm for determining if the nonadaptive broadcast time is at most a given constant t. We use the term target broadcast time for such a t. As, from Theorem 3.2, we know that $B_{\mathrm{na}}(T) \le B_{\mathrm{cl}}(T) + \lceil \frac{\mathrm{diam}(T)}{2} \rceil \le (n-1) + \lceil \frac{n-1}{2} \rceil$, we can use this algorithm in a binary search to determine the nonadaptive broadcast time $B_{\mathrm{na}}(T)$.

Let T be a tree with an arbitrarily chosen root r and let t be a target broadcast time for T. We describe a dynamic programming algorithm which examines subtrees beginning at the leaves and ending at the root. To describe what happens in the subtree T_v rooted at v, we introduce parameters down and out. These parameters are measurable for any given scheme σ. The parameter out is the maximum, over all originators in the subtree, of the time at which the message leaves the subtree (goes across the edge from v to v's parent) under σ. The parameter down is the time required to broadcast within the subtree from originator v under σ. A subscheme σ' of σ consists of some subset of the lists in σ.

Formally, we will write down^{σ}_v for the down parameter for T_v under σ (and similarly out^{σ}_v). If σ is clear from context, we omit it from the notation, and we also abbreviate down_{v_i} to down_i (and similarly for out).

For each vertex v, we are interested in the set of pairs $(\mathrm{down}, \mathrm{out})$ that can be realized by some scheme σ that achieves the target broadcast time t. Ultimately, if there is a realizable pair $(\mathrm{down}_r, \mathrm{out}_r)$ for root r, then the target time t can be achieved by some scheme σ.

Our algorithm computes the set of possible $(\mathrm{down}, \mathrm{out})$ pairs for each vertex beginning with the leaves. At the leaves, the only realizable pair is $(0,1)$.

At some arbitrary nonleaf vertex v with children $v_1, v_2, \ldots, v_d$, we assume that the set of realizable pairs for each v_i $(1 \le i \le d)$ has been computed. We now consider each potential pair for v; as the values of down and out are each trivially bounded by $O(n)$, there are $O(n^2)$ potential pairs to consider. Considering each pair in turn, then, we want to determine if the given pair is realizable by some scheme σ.

An easy, brute force, way to compute the set of realizable pairs for vertex v would be to consider all ways of constructing a list L_v of the edges incident on v and for each such list to compute the corresponding pairs.

Given a choice of the list for vertex v, let $l_v(j)$ denote the position of child v_j in the list, and let $l_v(0)$ denote the position of the parent of v in the list if v is not the root. Combining this choice of the list for vertex v with any choice of subschemes for the subtrees rooted at the children of v, we obtain a subscheme for the subtree rooted at v. In fact, to determine the realizable pairs for v, one does not have to know the details of the child subschemes but merely the realizable pairs for each child. For any set of choices of realizable pairs for the children, we can compute a $(\mathrm{down}, \mathrm{out})$ pair as follows:

$$\mathrm{down} = \max_{1 \le i \le d} \{ l_v(i) + \mathrm{down}_i \}$$

$$\mathrm{out} = l_v(0) + \max_{1 \le i \le d} \{ \mathrm{out}_i \}$$

This pair $(\mathrm{down}, \mathrm{out})$ is realizable if the subscheme satisfies the target time bound, or $\max_{\substack{1 \le i,j \le d \\ i \ne j}} \{ \mathrm{out}_i + l_v(j) + \mathrm{down}_j \} \le t$ and $\mathrm{out} \le t$.

At this point, we can construct an algorithm to compute the nonadaptive broadcast time of the tree (see Algorithm 1).

Algorithm 1.

1. Perform binary search to find optimal value of t between 1 and $3n/2$:
    2. **For** each vertex v in a postorder (leaf to root) traversal of T:
        3. **For** each possible choice of the list L_v:
            4. **For** each choice of realizable pairs $(\text{down}_i, \text{out}_i)$ for each child v_i of v
                5. Compute a (down, out) pair and record it if it is realizable.

Unfortunately, this algorithm is not polynomial. In particular, if Δ is the maximum degree of T, then step 3 represents $\Delta!$ iterations of steps 4 and 5. Also, because there can be a quadratic number of realizable pairs at any vertex, step 4 represents $(n^2)^\Delta$ iterations of step 5.

To obtain a more efficient algorithm, we redefine realizability so that a pair (down, out) is called realizable at v if there is some scheme σ such that the down and out parameters measured at v under σ are at most the corresponding values in the pair. Let (down, out) and (down$'$, out) with down $\leq$ down$'$ be two pairs realizable at v. In any scheme σ for T that achieves the target time t using a subscheme σ' at v corresponding to the pair (down$'$, out), we may replace σ' by a subscheme corresponding to the pair (down, out) to get another scheme for T that achieves the target time t.

So, at each vertex v for each value of out_v, consider the smallest value of down for which there is a realizable pair. We will denote this value of down by $\text{down}_v(\text{out}_v)$. By the above discussion, the values of $\text{down}_v(\text{out}_v)$ for each value of out_v are sufficient to characterize the realizable pairs for the subtree T_v.

Thus, at each vertex v for each value of out_v we need only consider the pair with the smallest value of down. This reduces the number of realizable pairs at any child from quadratic to linear in t. In turn, this reduces the number of iterations of step 4 from $(n^2)^\Delta$ to n^Δ. This approach still does not lead to a polynomial time algorithm. To achieve polynomial time, we shift our focus to determining whether or not a given pair is realizable at v rather than trying all possible choices of lists for v and all possible pairs at each v_i. It suffices to compute $\text{down}_v(\text{out}_v)$ for each value of out_v in place of steps 3, 4, and 5 of Algorithm 1, leading to Algorithm 2.

It remains to show how to compute $\text{down}_v(\text{out}_v)$. To do this, we first examine the local structure of an optimal scheme.

Let v be a nonleaf vertex with children $v_1, v_2, \ldots, v_d$. Given a particular value for out_v, there is a scheme σ (possibly more than one) that yields the value $\text{down}_v^\sigma = \text{down}_v(\text{out}_v)$. Under scheme σ, each subtree T_i has particular values down_v^σ and out_v^σ. We assume that the subtrees are indexed such that $\text{down}_1^\sigma \geq \text{down}_2^\sigma \geq \cdots \geq \text{down}_d^\sigma$.

Consider the values $\text{out}_1^\sigma, \text{out}_2^\sigma, \ldots, \text{out}_d^\sigma$. We examine two cases depending on whether the maximum of these values is repeated.

CASE 1. There exist at least two distinct i and j such that $\text{out}_i^\sigma = \text{out}_j^\sigma = \max_{1 \leq k \leq d}\{\text{out}_k^\sigma\}$.

In this case, we show that one of the possible schemes σ that yields $\text{down}_v(\text{out}_v)$ is such that $l_v(1) < l_v(2) < \cdots < l_v(d)$, that is, the children appear in order in L_v. Suppose this was not the case. Then, under σ there exists $1 \leq p < q \leq d$ such that $l_v(p) > l_v(q)$. We assume that σ is an optimal scheme (satisfying the target time bound t) with the minimum number of such inversions (p, q pairs). By our case assumptions, we furthermore assume that $p \neq i$. Consider the scheme σ^* obtained by starting with σ and swapping the positions of v_p and v_q in L_v giving L_v^*. Note that σ^* has fewer inversions. We now show that σ^* also yields $\text{down}_v(\text{out}_v)$, a contradiction. In the analysis below, we use $l_v(k)$ to denote the position of v_k in L_v but not in L_v^*.

In the broadcasts corresponding to σ and σ^*, the only differences are the times at which v sends to two of its children. Consequently, the time taken for a message to get from any vertex to v is the same in both schemes. It also follows that the value of out_v is the same for both schemes.

Under σ^*, the time for a message from v to reach a vertex in any T_k, where k is not p or q, is unchanged. The maximum time for such a message to reach a vertex of T_p is $l_v(q) + \text{down}_p < l_v(p) + \text{down}_p \leq \text{down}_v^\sigma$. The maximum time for such a message to reach a vertex of T_q is $l_v(p) + \text{down}_q \leq l_v(p) + \text{down}_p$ by our assumption that the subtrees are ordered by their down values; this is

Algorithm 2.

1. Perform binary search to find optimal value of t between 1 and $3n/2$:
    2. **For** each vertex v in a postorder (leaf to root) traversal of T:
        3. **For** each possible value of out_v between 1 and t:
            4. Compute $\text{down}_v(\text{out}_v)$.

also at most down_v^σ. From these cases, we have shown that $\text{down}^{\sigma^*} \leq \text{down}^\sigma$.

It remains to show that σ^* satisfies the target time bound t. The only message propagation times that are different in σ and σ^* are those passing through v with destination vertex in T_p or T_q. For destinations in T_p, these times are decreased under σ^*. Now consider destinations in T_q. The propagation time from any originator in any subtree T_a to a destination in T_q is bounded by $\text{out}_a^\sigma + l_v(p) + \text{down}_q^\sigma \leq \text{out}_i^\sigma + l_v(p) + \text{down}_p^\sigma$. This bound is at most t as it describes the time taken for some message broadcast from the subtree T_i to subtree T_p under σ, recalling that we earlier assumed that $p \neq i$. Thus, σ^* completes in time T.

CASE 2. There is exactly one i such that $\text{out}_i^\sigma = \max_{1 \leq k \leq d}\{\text{out}_k^\sigma\}$.

We show that one of the possible schemes σ that yields $\text{down}_v(\text{out}_v)$ is such that both $l_v(1) < l_v(2) < \cdots < l_v(i-1) < l_v(i+1) < \cdots < l_v(d)$ and $l_v(i-1) < l_v(i)$, that is, the children appear in order in L_v except v_i which may appear later. Suppose this was not the case. Then, under σ, there exists $1 \leq p < q \leq d$ such that $l_v(p) > l_v(q)$ and $p \neq i$. Further, we assume that σ is an optimal scheme with the minimum number of such inversions.

Consider the scheme σ^* obtained by starting with σ and swapping the positions of v_p and v_q in L_v. Note that σ^* has fewer inversions having $p \neq i$. We now show that σ^* also yields $\text{down}_v(\text{out}_v)$, a contradiction. The analysis proceeds as in Case 1.

We are now ready to state the algorithm that computes $\text{down}_v(\text{out})$. This algorithm is steps 5 through 21 of Algorithm 3. The array DOWN[v, out] stores the value $\text{down}_v(\text{out})$. Because of our redefinition of realizable, the scheme realizing the down value $\text{down}_v(\text{out})$ may have out_v less than the upper bound out. We store the actual out_v in the array OUT[v, out].

Consider any vertex v in T. If v is a leaf, then $\text{down}_v(\text{out}) = 0$ for any value of out and the array OUT[v, out] stores $\text{out}_v = 1$. These values are correctly computed in steps 5 and 6.

Otherwise, v's parent appears in position $l_v(0)$ in v's list L_v in an optimum scheme σ. As we can not predetermine which value of $l_v(0)$ leads to $\text{down}_v(\text{out}_v)$, we simply try each possible value (step 7). Steps 8–10 re-order the children of v to be consistent with our analysis above. If there is an optimal scheme σ such that case 1 occurs at v, then we compute the correct $\text{down}_v(\text{out} - v)$ in steps 11–14. If there is an optimal scheme σ such that case 2 occurs at v there is a unique maximum out_i ($i \in \{1, 2, \ldots, d\}$) and steps 16–20 are executed. The steps 17–20 try all of the possible lists L_v as described in case 2 and the optimal value of $\text{down}_v(\text{out})$ will be obtained.

Subroutine DOWNVALUE shown in Algorithm 4 computes the value of down_v^σ for a scheme σ having list L_v at vertex v and subschemes for children v_i, $i \in \{1, 2, \ldots, d\}$) characterized by down_i and out_i. If this scheme violates the target time bound t, the subroutine returns infinity. A naive computation of DOWNVALUE would check each pair of children v_i and v_j against the target time bound as in steps 3 and 4. By choosing a child v_i with maximum out_i value in step 1, we reduce the number of checks to linear in the number of children. This yields a time bound of $O(\Delta)$ for one iteration of the subroutine.

The loop of steps 17–20 has $O(\Delta)$ iterations of $O(\Delta)$ time, giving $O(\Delta^2)$ total time. Steps 15, 16 take $O(\Delta)$ time, giving $O(\Delta^2)$ total time for steps 15–20. Steps 11, 12 take time $O(\Delta)$ as do steps 13, 14. Step 10 takes $O(\Delta \log \Delta)$ time. Steps 8, 9 take $O(\Delta)$ time. Thus, the loop of steps 7–20 which is executed $O(\Delta)$ times requires a total of $O(\Delta^3)$ time. As steps 6 and 21 are both $O(1)$ time, the entire conditional of steps 5–21 takes $O(\Delta^3)$ time. Step 4 is constant time and the loop of steps 3–21 has $O(n)$ iterations [as t is $O(n)$] of $O(\Delta^3)$ time giving a total of $O(\Delta^3 n)$ time. The loop of steps 2–21 has $O(n)$ iterations giving a total of $O(\Delta^3 n^2)$ time. Finally, the binary search has $O(\log n)$ iterations giving a total time for Algorithm 3 of $O(\Delta^3 n^2 \log n)$ time.

Our objective was to find a polynomial algorithm to determine the nonadaptive broadcast time of a tree T. We believe that the actual polynomial can be improved by at least a factor of Δ.

5. LARGE TREES WITH SMALL NONADAPTIVE BROADCAST TIME

In this section, we consider the following problem: given a time t, determine a tree T_{na}^t with the largest n having nonadaptive broadcast time t. We call these trees nonadaptive broadcast trees and let N_{na}^t denote this largest n. We also use the phrase minimum scheme to denote a nonadaptive broadcast scheme on such a T that completes in time t and denote such a scheme $\hat{\sigma}$. To avoid trivialities, we insist that $t \geq 2$ and consider only trees of at least $n \geq 3$ vertices.

The subtree below a vertex v in a rooted tree can be described by the two parameters out_v and down_v. In a rooted tree T with a given nonadaptive broadcast scheme σ, we use the term branch at vertex v to refer to the subtree rooted at (nonroot) vertex v together with the lists L_u from σ for each vertex u in the subtree. Note that these lists include an entry in L_v for the parent of v. For branch R at vertex v, we use $\text{out}(R)$ and $\text{down}(R)$ analogously to out and down as described in Section 4. In particular, $\text{out}(R)$ is the max over all originators in R of the time at which the message being broadcast is sent from v to its parent and $\text{down}(R)$ is the time required to broadcast from v in R.

We extend the notion of broadcast tree to that of a broadcast tree under constraints Γ by which we mean a tree with the most vertices (and associated nonadaptive broadcast scheme) satisfying the set of constraints Γ. We denote such a tree as T_{na}^Γ and its size as N_{na}^Γ. We similarly define a broadcast branch under constraints Γ, R_{na}^Γ with size M_{na}^Γ. The difference between a broadcast branch and a broadcast tree in this context is that the list for the root of a broadcast branch includes an entry for the parent of the root whereas the list of the root of a broadcast tree does not.

Algorithm 3.

1. Perform binary search to find optimal value of t between 1 and $3n/2$:
   2. **For** each vertex v in a postorder (leaf to root) traversal of T:
      3. **For** each possible value of out_v between 1 and t:
         4. $MINDOWN \leftarrow \infty$
         5. **If** v is a leaf
            Then:
               6. $\text{DOWN}[v, \text{out}_v] \leftarrow 0$
                  $\text{OUT}[v, \text{out}_v] \leftarrow 1$
            Else:
               7. **For** every value $l_v(0) \in \{1, 2, \ldots, d+1\}$
                  (where d is the number of children of v):
                  8. **For** each child v_i where $i \in \{1, 2, \ldots, d\}$:
                     9. $\text{down}_i \leftarrow \text{DOWN}[v_i, \text{out}_v - l_v(0)]$
                        $\text{out}_i \leftarrow \text{OUT}[v_i, \text{out}_v - l_v(0)]$
                  10. Re-index the children of v so that $\text{down}_1 \geq \text{down}_2 \geq \cdots \geq \text{down}_d$
                  11. **For** each child v_i where $i \in \{1, 2, \ldots, d\}$:
                     12. **If** $l_v(0) > i$ **Then** $l_v(i) \leftarrow i$ **Else** $l_v(i) \leftarrow i + 1$
                  13. **If** $\text{DOWNVALUE}(v, L_v) < MINDOWN$
                     Then:
                        14. $MINDOWN \leftarrow \text{DOWNVALUE}(v, L_v)$
                            $MINOUT \leftarrow \text{out}_v$
                  15. **If** there is a unique maximum out_i ($i \in \{1, 2, \ldots, d\}$)
                     Then:
                        16. Let $MAXI$ be this unique i.
                        17. **For** each value of j between $= MAXI + 1$ and d:
                           18. Swap the positions of v_{j-1} and v_j in L_v
                           19. **If** $\text{DOWNVALUE}(v, L_v) < MINDOWN$
                              Then:
                                 20. $MINDOWN \leftarrow \text{DOWNVALUE}(v, L_v)$
                                     $MINOUT \leftarrow \text{out}_i + l_v(0)$
               21. $\text{DOWN}[v, \text{out}_v] \leftarrow MINDOWN$
                   $\text{OUT}[v, \text{out}_v] \leftarrow MINOUT$

In what follows we describe the size and structure of a nonadaptive broadcast tree. We do this by first identifying a family $\mathcal{G}_t$ of sets of constraints such that

$$N^t_{\text{na}} = \max_{\Gamma \in \mathcal{G}_t} N^{\Gamma}_{\text{na}}.$$

We next examine the structure of T^{Γ}_{na} and thereby show that

$$N^{\Gamma}_{\text{na}} = 1 + \sum_{\Gamma' \in \mathcal{F}(\Gamma)} M^{\Gamma'}_{\text{na}}$$

where $\mathcal{F}(\Gamma)$ is a family of sets of constraints dependent only on Γ. Note that at this point we will have reduced the problem of finding the size of a nonadaptive broadcast tree to that of finding the sizes of nonadaptive broadcast branches under certain constraints.

To determine the size $M^{\Gamma'}_{\text{na}}$ of a nonadaptive broadcast branch under constraints Γ', we will perform a similar analysis. We identify a family of sets of constraints $\mathcal{G}'(\Gamma')$ such that

$$M^{\Gamma'}_{\text{na}} = \max_{\Lambda \in \mathcal{G}'(\Gamma')} M^{\Lambda}_{\text{na}}.$$

We next examine the structure of R^{Λ}_{na} and thereby show that

$$M^{\Lambda}_{\text{na}} = 1 + \sum_{\Lambda' \in \mathcal{F}'(\Lambda)} M^{\Lambda'}_{\text{na}}$$

Algorithm 4.

$\text{DOWNVALUE}(v, L_v)$
1. Let v_i be a child of v with maximum out_i
2. **For** every value $j \in \{1, 2, \ldots, d\} - \{i\}$:
   3. **If** $\text{out}_i + l_v(j) + \text{down}_j > t$ or $\text{out}_j + l_v(i) + \text{down}_i > t$
   4. **Then** return ∞
5. Let v_k be a child of v with maximum $l_v(k) + \text{down}_k$
6. return $l_v(k) + \text{down}_k$

where $\mathcal{F}'(\Lambda)$ is a family of sets of constraints dependent only on Λ. Furthermore, the sets of constraints Λ' will have the same form as the sets Γ' and, thus, we will have a recursive formulation for $M_{\text{na}}^{\Gamma'}$. We now proceed to fill in the details of this general argument.

In the rest of this section, we consider a tree T with a scheme $\hat{\sigma}$ and root r. Also, we assume that r is not a leaf. We let $\delta > 1$ be the degree of the root r and $v_1, v_2, \ldots, v_\delta$ be the neighbors of r in the order that they appear in the list L_r of scheme $\hat{\sigma}$. We denote the branches at $v_1, v_2, \ldots, v_\delta$ as $R_1, R_2, \ldots, R_\delta$, respectively. Let firstbranch(T) be the least index of a branch R_i with maximum out(R_i) value and let secondbranch(T) be the least index of a branch R_j with maximum out(R_j) value over the remaining branches.

We define the set of tree constraints

$$\Gamma(t,i,j,\omega_1,\omega_2) = \left\{ \begin{array}{l} \text{the nonadaptive broadcast time of } T \\ \quad \text{under } \sigma \text{ is at most } t \\ i = \text{firstbranch}(T) \\ j = \text{secondbranch}(T) \\ \omega_1 = \text{out}(R_i) \\ \omega_2 = \text{out}(R_j) \end{array} \right\}$$

and we let

$$\mathcal{G}_t = \{\Gamma(t,i,j,\omega_1,\omega_2) \mid 1 \le i,j < t, i \ne j, 1 \le \omega_2 \le \omega_1 < t\}.$$

Lemma 5.1. $N_{\text{na}}^t = \max_{\Gamma \in \mathcal{G}_t} N_{\text{na}}^\Gamma$.

Proof. Consider the tree T_{na}^t. It completes broadcast in time at most t. Let $\hat{\imath} = \text{firstbranch}(T_{\text{na}}^t)$, $\hat{\jmath} = \text{secondbranch}(T_{\text{na}}^t)$, $\hat{\omega}_1 = \text{out}(R_{\hat{\imath}})$, and $\hat{\omega}_2 = \text{out}(R_{\hat{\jmath}})$. T_{na}^t satisfies the constraint set $\Gamma(t,\hat{\imath},\hat{\jmath},\hat{\omega}_1,\hat{\omega}_2)$ and, thus, $T_{\text{na}}^t = T_{\text{na}}^\Gamma$ for $\Gamma = \Gamma(t,\hat{\imath},\hat{\jmath},\hat{\omega}_1,\hat{\omega}_2)$.

As δ must be less than t for the broadcast to complete by time t, we have that $1 \le \hat{\imath}, \hat{\jmath} < t$. By definitions of firstbranch and secondbranch, $\hat{\imath} \ne \hat{\jmath}$. By the same definitions, $1 \le \hat{\omega}_2 \le \hat{\omega}_1$ and as the scheme completes by time t and r is not a leaf, we get $\hat{\omega}_1 < t$. So, $\Gamma \in \mathcal{G}_t$ and thus N_{na}^t is a candidate for the maximum.

For any $\Gamma \in \mathcal{G}_t$, T_{na}^Γ completes its broadcast by time t by the first constraint in Γ. As T_{na}^t was defined as the maximum-sized tree that completes by time t, N_{na}^t must be the maximum. ∎

Lemma 5.2. *Let T be a nonadaptive broadcast tree under constraints $\Gamma(t,i,j,\omega_1,\omega_2)$. Then, $\delta = t - \omega_1$.*

Proof. Otherwise, we can add a vertex pendant to r increasing both δ and the size of the tree without violating the constraints. ∎

Lemma 5.3. *Let T be a nonadaptive broadcast tree under constraints $\Gamma(t,i,j,\omega_1,\omega_2)$. Then,*

$$\begin{array}{ll} \text{down}(R_k) \le t - k - \omega_1 & \text{for } 1 \le k \le \delta, k \ne i \\ \text{down}(R_i) \le t - i - \omega_2 & \\ \text{out}(R_k) \le \omega_2 & \text{for } 1 \le k \le \delta, k \ne i,j \\ \text{out}(R_i) = \omega_1 & \\ \text{out}(R_j) = \omega_2 & \end{array}$$

Proof. Consider a broadcast from an originator within some branch R_l. For $k \ne l$, the message will reach v_k no later than time out$(R_l)+k$ and will reach all vertices of R_k by time out$(R_l) + k + \text{down}(R_k)$. Thus, for $1 \le k \le \delta$ and $k \ne l$, down$(R_k) \le t - k - \text{out}(R_l)$.

As the originator branch R_l was chosen arbitrarily, we have that down$(R_k) \le t - k - \text{out}(R_l)$ for $1 \le k, l \le \delta$ and $k \ne l$. This gives $\delta - 1$ constraints on each down(R_l), the tightest of which is when out(R_l) is maximized. When $k = i$, the maximum out(R_l) is ω_2, and when $k \ne i$, the maximum is ω_2. Thus, down$(R_k) \le t - k - \omega_1$ for $k \ne i$ and down$(R_i) \le t - i - \omega_2$. We now have an upper bound on the value of down for every branch.

To obtain an upper bound on out for every branch, we simply need note that out(R_i) and out(R_j) are part of the constraint set and the remaining values are no more than out$(R_j) = \omega_2$. ∎

Motivated by these lemmas, we define the set of branch constraints for branch R

$$\Gamma'(t,d,o) = \left\{ \begin{array}{l} \text{the nonadaptive broadcast time of } R \text{ under} \\ \quad \sigma \text{ is at most } t \\ d \le \text{down}(R) \\ o \le \text{out}(R) \end{array} \right\}$$

and we let

$$\mathcal{F}(\Gamma) = \left\{ \Gamma'(t,d_k,o_k) \mid \begin{array}{l} d_k = t - k - \omega_1 \\ \quad \text{for } 1 \le k \le t - \omega_1 \text{ and } k \ne i \\ d_i = t - i - \omega_2 \\ o_k = \omega_2 \\ \quad \text{for } 1 \le k \le t - \omega_1 \text{ and } k \ne i \\ o_i = \omega_1 \end{array} \right\}.$$

Note that the five parameters $t, i, j, \omega_1, \omega_2$ are inherent in the definition $\Gamma = \Gamma(t,i,j,\omega_1,\omega_2)$.

In contrast to $\mathcal{G}_t$, which contained many sets of constraints to maximize over, the family $\mathcal{F}(\Gamma)$ has a small number of sets of constraints (one for each child of the root) and we sum over this set.

Lemma 5.4. $N_{\text{na}}^\Gamma = 1 + \sum_{\Gamma' \in \mathcal{F}(\Gamma)} M_{\text{na}}^{\Gamma'}$

Proof. By Lemma 5.2, T_{na}^Γ consists of a root vertex with $t - \omega_1$ subtrees. The root contributes 1 to N_{na}^Γ and subtree k contributes $M_{\text{na}}^{\Gamma'}$ with $\Gamma' = \Gamma'(t,d_k,o_k)$ where d_k and o_k are specified by Lemma 5.3. Note that Lemma 5.3 contains inequalities where the definition of $\mathcal{F}(\Gamma)$ uses

equalities; the required inequalities are present in the definition of $\Gamma'(t,d,o)$. ∎

Combining Lemmas 5.1 and 5.4, we have reduced the problem of finding the size of a nonadaptive broadcast tree to that of finding the sizes on nonadaptive broadcast branches under constraints on t, down, and out:

Corollary 5.4.1. $N^t_{\text{na}} = \max_{\Gamma \in \mathcal{G}_t}\{1 + \sum_{\Gamma' \in \mathcal{F}(\Gamma)} M^{\Gamma'}_{\text{na}}\}$.

To find these sizes (i.e., the values of $M^{\Gamma'}_{\text{na}}$), we do a similar analysis.

We consider a branch R at vertex s with a scheme $\hat{\sigma}$. Also, we let degree$(R) > 1$ be the degree of s and $v_1, v_2, \ldots, v_{\text{degree}(R)}$ be the neighbors of s in the order that they appear in the list L_r of scheme $\hat{\sigma}$. We assume that the parent of s is in this list and let parent(R) be its index. We denote the sub-branch at v_k, $k \neq$ parent(R), as R_k. Let firstbranch(R) be the least index of a sub-branch R_i with maximum out(R_i) value and let secondbranch(B) be the least index of a sub-branch R_j with maximum out(R_j) value over the remaining sub-branches.

Analogously to $\Gamma(t,i,j,\omega_1,\omega_2)$, we define the set of branch constraints for branch R. The situation here is more complicated as we require several more parameters and must consider three cases depending on the degree of s.

In the general case, the degree of s is at least three. Here, we define

$$\Lambda_3(t,i,j,p,\delta,d,\omega_1,\omega_2) = \left\{\begin{array}{l}\text{the nonadaptive broadcast time of } R \text{ under } \sigma \text{ is at most } t\\ i = \text{firstbranch}(R)\\ j = \text{secondbranch}(R)\\ p = \text{parent}(R)\\ \delta = \text{degree}(R)\\ d = \text{down}(R)\\ \omega_1 = \text{out}(R_i)\\ \omega_2 = \text{out}(R_j)\end{array}\right\}$$

and we let $\mathcal{G}'_3(\Gamma') = \{\Lambda_3(\tilde{t},i,j,p,\delta,\tilde{d},\omega_1,\omega_2) \mid \tilde{t} = t,\ 1 \le i,j,p \le \delta, i \neq j, i \neq p, j \neq p, 0 \le \tilde{d} \le d, \delta \le t-\omega_1+1, 1 \le \omega_2 \le \omega_1 \le o < t\}$. The parameters t,d,o come from the definition $\Gamma' = \Gamma'(t,d,o)$.

When the degree of s is one, there is only one possibility for the branch structure, so we define $\Lambda_1 = \{\delta = 1\}$ and we let $\mathcal{G}'_1(\Gamma') = \{\Lambda_1\}$.

When the degree of s is two, we define

$$\Lambda_2(t,i,p,d,\omega_1) = \left\{\begin{array}{l}\text{the nonadaptive broadcast time of } R \text{ under } \sigma \text{ is at most } t\\ i = \text{firstbranch}(R)\\ p = \text{parent}(R)\\ d = \text{down}(R)\\ \omega_1 = \text{out}(R_i)\end{array}\right\}$$

and we let $\mathcal{G}'_2(\Gamma') = \{\Lambda_2(\tilde{t},i,p,\tilde{d},\omega_1) \mid \tilde{t} = t,\ 1 \le i,p \le 2, i \neq p, 0 \le \tilde{d} \le d, 1 \le \omega_1 \le o < t\}$.

Finally, we compose our family of sets of constraints by collecting the sets of constraints from all three cases: $\mathcal{G}'(\Gamma') = \cup_{i=1,2,3}\, \mathcal{G}'_i(\Gamma')$.

Unlike the analogous $\mathcal{G}_t$, the family $\mathcal{G}'(\Gamma')$ is not clearly composed of only feasible sets of constraints. To guard against this possibility, we introduce a feasibility constraint on the maximum in the following lemma that was not present in Lemma 5.1.

Lemma 5.5. $M^{\Gamma'}_{\text{na}} = \max_{\text{feasible } \Lambda \in \mathcal{G}'(\Gamma')} M^{\Lambda}_{\text{na}}$.

Proof. In this proof, we consider only the general elements of $\mathcal{G}'(\Gamma')$, that is, those elements of $\mathcal{G}'(\Gamma')$ that have $\delta \ge 3$. The special cases for $\delta = 1, 2$ can be handled similarly.

Consider the branch $R^{\Gamma'}_{\text{na}}$. It completes broadcast in time at most t. Let $\hat{\imath} = \text{firstbranch}(R^{\Gamma'}_{\text{na}})$, $\hat{\jmath} = \text{secondbranch}(R^{\Gamma'}_{\text{na}})$, $\hat{p} = \text{parent}(R^{\Gamma'}_{\text{na}})$, $\hat{\delta} = \text{degree}(R^{\Gamma'}_{\text{na}})$, $\hat{d} = \text{down}(R^{\Gamma'}_{\text{na}})$, $\hat{\omega}_1 = \text{out}(R_{\hat{\imath}})$, and $\hat{\omega}_2 = \text{out}(R_{\hat{\jmath}})$. $R^{\Gamma'}_{\text{na}}$ satisfies the constraint set $\Lambda(t,\hat{\imath},\hat{\jmath},\hat{p},\hat{\delta},\hat{d},\hat{\omega}_1,\hat{\omega}_2)$ and, thus, $R^{\Gamma'}_{\text{na}} = R^{\Lambda}_{\text{na}}$ for $\Lambda = \Lambda(t,\hat{\imath},\hat{\jmath},\hat{p},\hat{\delta},\hat{d},\hat{\omega}_1,\hat{\omega}_2)$.

By definition, we have that $1 \le \hat{\imath},\hat{\jmath},\hat{p} \le \hat{\delta}$. By the definitions of firstbranch, secondbranch, and parent, we have $\hat{\imath} \neq \hat{\jmath}$, $\hat{\imath} \neq \hat{p}$, and $\hat{\jmath} \neq \hat{p}$. By the same definitions, $1 \le \hat{\omega}_2 \le \hat{\omega}_1$. Furthermore, $\hat{\delta} \le t - \hat{\omega}_1 + 1$ as if $\hat{\delta} \ge t - \hat{\omega}_1 + 2$ then there is some broadcast originating in R_i that takes time $\hat{\omega}_1$ to get to s and some neighbor of s not in R_i that receives the message $t - \hat{\omega}_1 + 1$ time units later, exceeding the time bound of t.

For any feasible $\Lambda \in \mathcal{G}'(\Gamma')$, by the definitions of $\Lambda(t,i,j,p,\delta,d,\omega_1,\omega_2)$ and $\mathcal{G}'(\Gamma')$, R^{Λ}_{na} completes its broadcast in time t, has down$(R^{\Lambda}_{\text{na}}) \le d$, and out$(R^{\Lambda}_{\text{na}}) \le o$. As $R^{\Gamma'}_{\text{na}}$ was defined as the maximum-sized branch that completes by time t with down at most d, and out at most o, $M^{\Gamma'}_{\text{na}}$ must be the maximum. ∎

Lemma 5.6. *Let R be a nonadaptive broadcast branch under constraints $\Lambda(t,i,j,p,\delta,d,\omega_1,\omega_2)$. Then,*

$$\begin{array}{ll}\text{down}(R_k) \le t - k - \omega_1 & \text{for all } k \neq i,p\\ \text{down}(R_i) \le t - i - \omega_2 & \\ \text{out}(R_k) \le \omega_2 & \text{for all } k \neq i,j,p\\ \text{out}(R_i) = \omega_1 & \\ \text{out}(R_j) = \omega_2 & \end{array}$$

Proof. The proof is essentially the same as that of Lemma 5.3 and is omitted. ∎

For the next definition, we reuse the set of branch constraints $\Gamma'(t,d,o)$ as defined before Lemma 5.4 and we let

$$\mathcal{F}'(\Lambda) = \{\Gamma'(t,d_k,o_k) \mid k \neq p \text{ and } 1 \le k \le \delta\}$$

$$\text{where } d_k = \min\left\{\left\{\begin{array}{ll}t-k-\omega_1 & \text{for } k \neq i,p\\ t-k-\omega_2 & \text{for } k = i\end{array}\right\}, d-k\right\}$$

TABLE 1. Small values of N^t_{na}.

t	N^t_{na}	t	N^t_{na}
0	1	13	104
1	2	14	148
2	2	15	221
3	3	16	325
4	4	17	467
5	6	18	656
6	8	19	963
7	12	20	1430
8	17	21	2086
9	24	22	2988
10	34	23	4212
11	51	24	6298
12	74	25	9286

$$o_k = \begin{Bmatrix} \omega_2 & \text{for } k \neq i,p \\ \omega_1 & \text{for } k = i \end{Bmatrix},$$

$$\delta = 1 \text{ when } \Lambda \in \mathcal{G}'_1(\Gamma'),$$

$$\text{and } \delta = 2 \text{ when } \Lambda \in \mathcal{G}'_2(\Gamma').$$

We now derive what appears to be a recursive formula for branch sizes under constraints. However, this formula is not directly recursive as the constraints Λ in $M^{\Lambda}_{\mathrm{na}}$ are not the same form as the constraints Λ'.

Lemma 5.7.

$$M^{\Lambda}_{\mathrm{na}} = \begin{Bmatrix} 1 + \sum_{\Gamma' \in \mathcal{F}'(\Lambda)} M^{\Gamma'}_{\mathrm{na}} & \Lambda \in \mathcal{G}'_3(\Gamma') \\ 1 & \Lambda \in \mathcal{G}'_1(\Gamma') \\ 1 + M^{\Gamma'(t,d-i,\omega_1-p)}_{\mathrm{na}} & \Lambda \in \mathcal{G}'_2(\Gamma') \end{Bmatrix}.$$

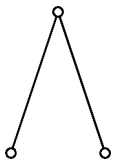

FIG. 1. The tree T^3_{na}.

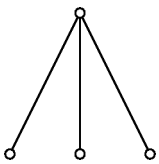

FIG. 2. The tree T^4_{na}.

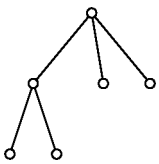

FIG. 3. The tree T^5_{na}.

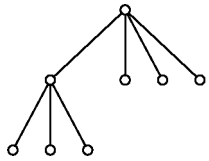

FIG. 4. The tree T^6_{na}.

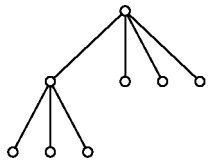

FIG. 5. The tree T^7_{na}.

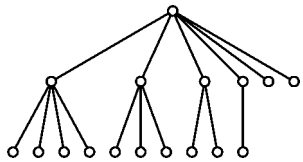

FIG. 6. The tree T^8_{na}.

Proof. If $\Lambda \in \mathcal{G}'_1(\Gamma')$, then $\delta = 1$ and the branch consists of a single vertex.

If $\Lambda \in \mathcal{G}'_2(\Gamma')$, then $\delta = 2$. Thus, the root of the branch has only one child c. The sub-branch at c has down value at most $d - i$ and out value at most $\omega_1 - p$. Note that the only feasible values for i and p are $i = 1, p = 2$ and $i = 2, p = 1$.

If $\Lambda \in \mathcal{G}'_3(\Gamma')$, then $\delta \geq 3$ and $R^{\Lambda}_{\mathrm{na}}$ consists of a root vertex with $\delta - 1$ subtrees. The root contributes 1 to $M^{\Lambda}_{\mathrm{na}}$ and subtree k contributes $M^{\Gamma'}_{\mathrm{na}}$ with $\Gamma' = \Gamma'(t, d_k, o_k)$. ∎

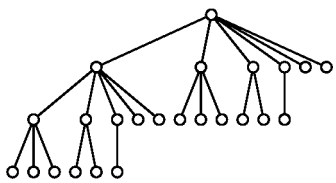

FIG. 7. The tree T^9_{na}.

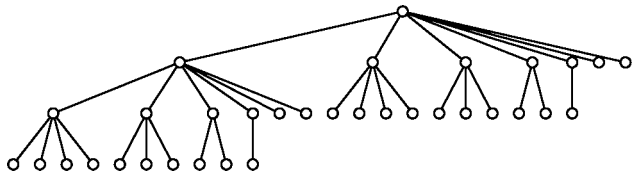

FIG. 8. The tree T^{10}_{na}.

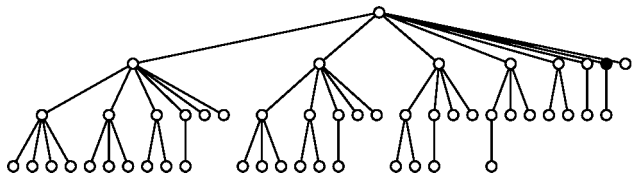

FIG. 9. The tree T^{11}_{na}.

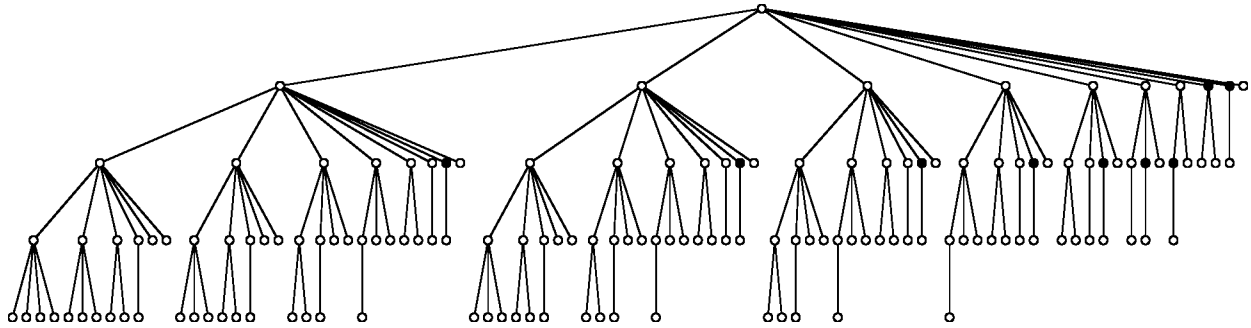

FIG. 10. Part of the tree T_{na}^{16}.

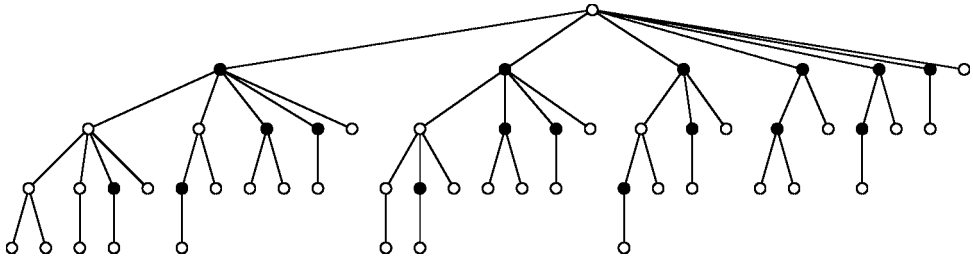

FIG. 11. Part of the tree T_{na}^{23}.

Combining Lemmas 5.5 and 5.7, we arrive at a truly recursive formula for branch sizes under constraints.

Theorem 5.8.

$$M_{\mathrm{na}}^{\Gamma'} = \max_{\text{feasible}\Lambda\in\mathcal{G}'(\Gamma')} \begin{Bmatrix} 1+\sum_{\Gamma'\in\mathcal{F}'(\Lambda)} M_{\mathrm{na}}^{\Gamma'} & \Lambda\in\mathcal{G}_3'(\Gamma') \\ 1 & \Lambda\in\mathcal{G}_1'(\Gamma') \\ 1+M_{\mathrm{na}}^{\Gamma'(t,d-i,\omega_1-p)} & \Lambda\in\mathcal{G}_2'(\Gamma') \end{Bmatrix}.$$

This result allows us to write a dynamic program to calculate $M_{\mathrm{na}}^{\Gamma'}$ for small values of t, d, and o. Corollary 5.4.1 shows us how to combine these values to obtain the values N_{na}^t for small t. These values are shown in Table 1 for $t \le 25$.

The largest trees T_{na}^t for time bounds t from 3 to 10 are shown in Figures 1–8. In the optimal broadcast schemes for these trees, every node informs its parent (if it has one) first, and then informs its children in left-to-right order.

However, in larger trees of this sequence, there are nodes that inform their children first and then their parent. In Figure 9, the tree T_{na}^{11} is shown. We follow the convention that the white-filled vertices inform their parent first, and the black-filled vertices inform their children first. With the same convention, in Figure 10, we show part of T_{na}^{16}. For this figure, we removed the left two subtrees of the root, to make it easier to discern the children-first nodes.

One interesting pattern is seen in the trees for $t = 5, 6, 9, 10$, and 14. If one removes the edge from the root to its leftmost subtree in any of these trees, then the remaining two subtrees are isomorphic.

In Figure 11, part of the tree T_{na}^{23} is shown. The 10 left subtrees of the root have been omitted. Note that in this tree, some black vertices have many children, and some paths to the root have more than one black vertex.

6. AN IMPROVED BOUND FOR NONADAPTIVE BROADCASTING IN GRAPHS

Considering general graphs rather than just trees, we note an improvement to another bound of Diks and Pelc [1] who showed that $B_{\mathrm{na}}(G) \le 3B_{\mathrm{cl}}(G)$ (Theorem 2.2 of Ref. [1]).

Theorem 6.1. *For any graph G, $B_{\mathrm{na}}(G) \le 3b_{\mathrm{cl}}(G) - 1$.*

Proof. Let c be a vertex in the classical broadcast center of G, and T be the spanning tree of G that consists of the edges which transmit the message in a classical broadcast on G from c. Note that $B_{\mathrm{na}}(G) \le B_{\mathrm{na}}(T)$, as this is true for any spanning tree of G. Then, by Theorem 3.3, $B_{\mathrm{na}}(T) \le \mathrm{diam}(T) + b_{\mathrm{cl}}(T) = \mathrm{diam}(T) + b_{\mathrm{cl}}(G)$. As $\mathrm{diam}(T) \le 2b_{\mathrm{cl}}(T) - 1 = 2b_{\mathrm{cl}}(G) - 1$, we have $B_{\mathrm{na}}(G) \le \mathrm{diam}(T) + b_{\mathrm{cl}}(G) \le 3b_{\mathrm{cl}}(G) - 1$. ∎

As $b_{\mathrm{cl}}(T) \le B_{\mathrm{cl}}(T)$, $3b_{\mathrm{cl}}(G) - 1 < 3B_{\mathrm{cl}}(G)$ and this is an improvement on the bound of Diks and Pelc for all graphs G.

REFERENCES

[1] K. Diks and A. Pelc, Broadcasting with universal lists, Networks 27 (1996), 183–196.

[2] P. Fraigniaud and E. Lazard, Methods and problems of communication in usual networks, Discr Appl Math 53 (1994), 79–133.

[3] H.A. Harutyunyan and P. Taslakian, "Orderly broadcasting in a 2D torus," 8th International Conference on Information Visualisation IV, London, UK, 2004, pp. 370–375.

[4] S.T. Hedetniemi, S.M. Hedetniemi, and A.L. Liestman, A survey of broadcasting and gossiping in communication networks, Networks 18 (1988), 319–349.

[5] J. Hromkovič, R. Klasing, B. Monien, and R. Peine, "Dissemination of information in interconnection networks (broadcasting and gossiping)," Combinatorial network theory, D.-Z. Du and D.F. Hsu (Editors), Kluwer, Dordrecht, The Netherlands, 1995, pp. 125–212.

[6] J. Hromkovič, R. Klasing, A. Pelc, P. Ružička, and W. Unger, Dissemination of information in communication networks: broadcasting, gossiping, leader election, and fault-tolerance, Springer, Berlin, 2005.

[7] J.-H. Kim and K.-Y. Chwa, Optimal broadcasting with universal lists based on competitive analysis, Networks 45 (2005), 224–231.

[8] A. Rosenthal and P. Scheuermann, "Universal rankings for broadcasting in tree networks," Proc 25th Allerton Conf Combinatorics, Control, Comput, Monticello, Illinois, USA, 1987, pp. 641–649.

[9] P.J. Slater, E.J. Cockayne, and S.T. Hedetniemi, Information dissemination in trees, SIAM J Comp 10 (1981), 692–710.