Parallel Processing Letters Vol. 8 No. 2 (1998) 149–159

MESSY BROADCASTING

HOVHANNES A. HARUTYUNYAN
School of Computing Science, Simon Fraser University
Burnaby, British Columbia, V5A 1S6, Canada

and

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

Received January 1997
Revised March 1998
Accepted by A. M. Farley & A. Proskurowski

ABSTRACT

In this note, we continue the study of messy broadcasting. We obtain exact values for the messy broadcasting time of complete graphs, paths, cycles, and complete d-ary trees. For hypercubes, we obtain exact values for messy broadcasting time under two of the models and present upper and lower bounds for the third model. We compare these times with (regular) broadcasting times in these graphs. We also present some simple bounds for arbitrary graphs which we use to compare the messy broadcasting times of cube-connected cycles, shuffle-exchange graphs, butterfly graphs, and DeBruijn graphs with their (regular) broadcasting times.

Keywords: broadcasting, gossiping, hypercubes, cube-connected cycles, shuffle-exchange graphs, butterfly graphs, DeBruijn graphs

1. Introduction

In the classical broadcast model, it is tacitly assumed that every vertex of the network broadcasts its message using an optimal scheme. In this model, it is assumed that either there is an omniscient being coordinating the actions of all of the vertices during the entire broadcasting process or that the vertices must have a coordinated set of protocols which are optimized for each originator. In the latter case, the vertices must have enough storage space for the schemes and they must be able to determine the originator of an incoming broadcast message.

In this paper, we assume that there is no omniscient being coordinating the actions and that the vertices do not know the state of the network, the originator, or the starting time of the broadcast message. Furthermore, they do not have coordinated protocols. That is, each vertex acts as an independent agent with a limited view of the network. The version of broadcasting considered here is called *messy* *broadcasting* and was introduced by Ahlswede, Haroutunian,[a] and Khatchatrian [1].

The following three models provide each vertex with slightly different views of their local neighborhood. We assume that each vertex can transmit a message to at most one of its neighbors in a given time unit, but can receive information from any number of its neighbors simultaneously.

Model M_1: At each unit of time, every vertex know the state of each of its neighbors: informed or uninformed. In this model, each informed vertex must transmit the broadcast message to one of its uninformed neighbors, if any, in each time unit.

Model M_2: Every informed vertex knows from which vertex (vertices) it received the broadcast message and to which neighbors it has sent the message. Thus, it knows that this vertex (or these vertices) are informed. In this model, each informed vertex must transmit the broadcast message to one of its neighbors other than the ones that it knows are informed, if any, in each time unit.

Model M_3: Every informed vertex knows to which neighbors it has sent the message. In this model, each informed vertex must transmit the broadcast message to one of its neighbors to which it has not yet sent the message, if any, in each time unit.

Contrary to the usual practice, we are concerned with the worst case performance of broadcast schemes in this model. We define the *broadcast time of vertex* u in graph G using model M_i, denoted $t_i(u)$, for $i = 1, 2, 3$, to be the maximum number of time units required to complete broadcasting from vertex u over all broadcast schemes for u. The *broadcast time of graph* G using model M_i, denoted $t_i(G)$, for $i = 1, 2, 3$, is the maximum broadcast time for any vertex u of G. That is, $t_i(G) = \max\{t_i(u) | u \in V\}$.

We use $deg_G(u)$ to denote the degree of vertex u in G, δ_G to denote the minimum degree of any vertex in G, Δ_G to denote the maximum degree of any vertex in G, and $d(G)$ to denote the diameter of G. When the meaning is clear from context, we may omit the subscript G.

We consider the problem of determining $t_i(G)$ for $i = 1, 2, 3$ in several families of graphs. In Section 2, we give exact values of $t_i(G)$, for $i = 1, 2, 3$, for complete graphs, paths, and cycles and some simple bounds on $t_i(G)$ for all graphs G. In Section 3, we determine the exact value of $t_i(G)$ for $i = 2, 3$ and provide bounds on $t_1(G)$ for hypercubes. In Section 4, we obtain exact values of $t_i(G)$, for $i = 1, 2, 3$, for complete d-ary trees. In Section 5, we summarize these results and compare the bounds obtained here to the known (regular) broadcasting times for the abovementioned graphs plus cube-connected cycles, shuffle-exchange graphs, butterflies, and de Bruijn graphs.

2. Messy Broadcasting in Simple Graphs

We use K_n, P_n, and C_n to denote the complete graph, path, and cycle on n

[a]Haroutunian is the previous spelling of Harutyunyan, the first author of this paper.

vertices, respectively.

Lemma 2.1 $t_1(K_n) = t_2(K_n) = t_3(K_n) = n - 1$.

Proof. In all 3 models, the originator must call its $n-1$ neighbors in times $1, 2, ..., n-1$, ensuring that the broadcast is complete no later than at time $n-1$. Thus, $t_i(K_n) \leq n-1$ for each i.

Without loss of generality, assume that the vertices of K_n are numbered $0, 1, ...,$ $n-1$ and that 0 is the originator. The following scheme is possible in each of the three models: At time i, all of the informed vertices call vertex i. This shows that $t_i(K_n) \geq n-1$ for each i. □

Lemma 2.2 *For any graph G on n vertices with m edges, $t_1(G) \leq m$, $t_2(G) \leq m$, and $t_3(G) \leq 2m-1$.*

Proof. We begin with M_3. Any single edge in the graph will be used at most twice in any messy broadcast scheme - each endpoint sending the message to its neighbor along the edge. However, the broadcast process ends when the last vertex is informed. (In fact, several vertices may learn the information simultaneously at that time.) Although the model requires that the vertex must call the neighbor from which it received the message, the broadcast is completed before this call is made. Thus, at most $2m-1$ calls are made before the broadcast is completed.

In the other models, once a vertex has received the message on a given edge, it does not send the message on the same edge. Thus, each edge is used at most once in these models. □

Let u be the originator of a messy broadcast and let $t_i(u, v)$, for $i = 1, 2, 3$, denote the maximum number of time units required to inform vertex v in any messy broadcasting scheme from originator u. It follows that $t_i(G) \leq \max\{t_i(u, v) | u, v \in V\}$.

Lemma 2.3 *Let v_0 be the originator of a messy broadcast in $G = (V, E)$. Let v_k be any vertex in V such that $v_0, v_1, v_2, \ldots, v_k$ is a shortest path from v_0 to v_k in G. Then, $t_3(v_0, v_k) \leq \sum_{i=0}^{k-1} deg(v_i)$ and $t_1(v_0, v_k) \leq t_2(v_0, v_k) \leq (\sum_{i=0}^{k-1} deg(v_i)) - (k-1)$.*

Proof. In any model M_3 messy broadcasting scheme for originator v_0, the originator will call all of its neighbors in some order. Thus, the call from v_0 to v_1 may occur as late as at time $deg(v_0)$. Once v_i is informed, it will also call all of its neighbors in some order and may call v_{i+1} as late as $deg(v_i)$ time units after it receives the message. The bound follows.

In models M_2 and M_1, vertex v_i, $i > 0$, may call its $deg(v_i) - 1$ neighbors other than v_{i-1} prior to calling v_{i+1}. Thus, vertices $v_1, \ldots, v_{k-1}$ may each make one less call than in model M_3. □

Corollary 2.1 *For any graph G, $t_3(G) \leq d(G)\Delta_G$ and $t_1(G) \leq t_2(G) \leq d(G)(\Delta_G - 1) + 1$.*

Proof. This follows immediately from Lemma 2.3 by considering vertices v_0 and v_k at distance $d(G)$. □

Lemma 2.4 *$t_1(P_n) = t_2(P_n) = n - 1$ and $t_3(P_n) = 2n - 3$.*

Proof. Since P_n has $n-1$ edges, $t_1(P_n) \leq n-1$, $t_2(P_n) \leq n-1$, and $t_3(P_n) = 2n-3$ follow from Lemma 2.2.

Let the vertices of P_n be $0, 1, ..., n-1$ such that vertex i is adjacent to $i-1$ and $i+1$ for $i = 2, ..., n-2$.

In M_3, consider the case of vertex 0 as the originator. At time 1, 0 calls its only neighbor 1. Subsequently, each vertex i first returns the message to $i-1$ at time $2i$ and then calls $i+1$ at time $2i+1$. The message arrives at $n-1$ at time $2n-3$, completing the broadcast. For other originators, messy broadcast completes no later than time $2n-3$.

In the other models, a worst case also occurs when 0 is the originator. At each time, the newly informed vertex has only one uninformed neighbor. The only possible scheme is for vertex i to call vertex $i+1$ at time $i+1$, completing the broadcast at time $n-1$. □

Lemma 2.5 *$t_1(C_n) = t_2(C_n) = \lceil \frac{n}{2} \rceil$ and $t_3(C_n) = n - 1$.*

Proof. As with the path, in models M_1 and M_2, there is only one possible scheme after the first call. (The first call can proceed in either direction around the cycle, but the rest of the scheme is completely determined.) The originator first calls one neighbor and then the other. All other vertices simply forward the message around the cycle, completing the broadcast at time $\lceil \frac{n}{2} \rceil$. No other scheme complies with the constraint that an uninformed neighbor must be called, thus, $t_1(C_n) = t_2(C_n) = \lceil \frac{n}{2} \rceil$.

We turn to the model M_3. At time 0, only the originator knows the information. This vertex must call an uninformed neighbor, so that 2 vertices know the information at time 1. Subsequently, any informed vertex may first call an informed neighbor, but must call an uninformed neighbor no more than 2 time units after it receives the message. (Note that at the end of the broadcast, there will be no uninformed neighbor.) Thus, the number of informed vertices at time t must be at least $t+1$, giving $t_3(C_n) \leq n-1$.

The following scheme shows that $t_3(C_n) \geq n-1$: Let the vertices be numbered $0, 1, ..., n-1$ with edges connecting i with $(i+1) \bmod n$ and $(i-1) \bmod n$. Without loss of generality, let 0 be the originator. At time 1, vertex 0 calls vertex $n-1$. At even times $2i > 1$, vertex $i-1$ calls vertex i and vertex $n-i$ calls (back to) vertex $(n-i+1) \bmod n$. At odd times $2i+1 > 1$, vertex i calls (back to) vertex $i-1$ and vertex $(n-i+1) \bmod n$ calls vertex $n-i$. □

3. Messy Broadcasting in Hypercubes

The binary hypercube of dimension d, denoted Q_d is a graph on 2^d vertices. These vertices are labelled with the binary strings of length d and two vertices are connected by an edge if and only if their labels differ in exactly one bit. We will denote the label of vertex u by $(u_1u_2...u_d)$ where $u_i \in \{0,1\}$ for $i = 1, 2, ..., d$. The edge between vertices u and v will be denoted (u, v).

Theorem 3.1 $t_3(Q_d) = \frac{d(d+1)}{2}$.

Proof. We begin by showing that $t_3(Q_d) \leq \frac{d(d+1)}{2}$. Consider any messy broadcast scheme for Q_d and let $v_0 = (00...0)$ be the originator. Let $f(i)$ denote the number of time units required to ensure that all vertices at distance i from the originator are informed. Note that the vertices at distance i from the originator v_0 have labels containing exactly i 1's. Such vertices are said to be on *level* i of the cube. Since the diameter of Q_d is d, $t_3(Q_d) \leq f(d)$.

Without loss of generality, let us assume that the last vertex informed on level i is $v_i = (00...011...1)$ and consider the i-dimensional subcube Q_i which contains exactly those vertices with labels beginning with $d-i$ 0's. Let $x(j)$ denote the time at which the first vertex of level j in this subcube is informed by the scheme. Beginning at time 1, the originator may make $d-i$ calls outside of Q_i but then must call an element, say u, of level 1 of Q_i. Thus, $x(1) \leq d-i+1$. Vertex u may call its $d-i$ neighbors in level 2 of Q_d outside of Q_i and the originator in time units $x(1)+1$, $x(1)+2$, ..., $x(1)+d-i+1$. It must then call a neighbor, say vertex x, in level 2 of Q_i, so $x(2) \leq x(1)+d-i+2$. Vertex x may call its $d-i$ neighbors in level 3 of Q_d outside of Q_i and its two neighbors in level 1 of Q_i in time units $x(2)+1$, $x(2)+2$, ..., $x(2)+d-i+2$. It must then call a neighbor in level 3 of Q_i, so $x(3) \leq x(2)+d-i+3$. Similarly, if v is the first element of level $j-1$ in Q_i to be informed, we argue that it must call a neighbor in level j of Q_i at time $x(j-1)+d-i+j$, giving $x(j) \leq x(j-1)+d-i+j$. Summing this inequality for $j = 1, ..., i$, we get $x(i) \leq \sum_{j=1}^{i}(d-i+j) = in - \frac{i(i-1)}{2}$. Letting $i = d$, we get $x(d) \leq d^2 - \frac{d(d-1)}{2} = \frac{d(d+1)}{2}$ and since level d has only one vertex, we get $t_3(Q_d) \leq f(d) \leq \frac{d(d+1)}{2}$.

To show that $t_3(Q_d) \geq \frac{d(d+1)}{2}$, we exhibit a scheme which requires this amount of time: When vertex $u = (u_1u_2...u_d)$ with i 1's receives the message, it first forwards the message to its neighbors with $i-1$ 1's and then to its neighbors with $i+1$ 1's. The neighbors on level $i-1$ have labels which differ from u's label by replacing a single 1 with a 0. The first neighbor on level $i-1$ which is sent the message is that neighbor whose label has a 0 corresponding to u's rightmost (last) 1. The second neighbor on level $i-1$ which is sent the message is that neighbor whose label has a 0 corresponding to u's next to last 1. The final neighbor on level $i-1$ which is sent the message is that neighbor whose label has a 0 corresponding to u's first 1. The first neighbor on level $i+1$ which is sent the message is that neighbor whose label has a 1 corresponding to u's leftmost (first) 0. The second neighbor on level $i+1$ which is sent the message is that neighbor whose label has a 1 corresponding to u's second 0. The final neighbor on level $i+1$ which is sent the message is that neighbor whose label has a 1 corresponding to u's last 0.

It is easy to show by induction that vertex $v_k = (11...100...0)$ of level k (that is, the label has k 1's preceding $d-k$ 0's) is informed by time $\frac{k(k+1)}{2}$ under this scheme. The result is trivially true for $(00...0)$, the originator. Assume that it is true for v_i and show that it is true for v_{i+1}. Then, v_i is informed at time $\frac{i(i+1)}{2}$ under this scheme. Vertex v_i will call all of its i level $i-1$ neighbors and will then call v_{i+1} at time $\frac{i(i+1)}{2} + i + 1 = \frac{(i+1)(i+2)}{2}$. It is not possible that v_{i+1} could be informed earlier than this time under this scheme. Noting that vertex v_i is the first vertex of level i which is informed and all other vertices on level i are informed strictly later, v_i is the first vertex to call any level $i+1$ vertex and v_{i+1} is the vertex that it calls. □

Theorem 3.2 $t_2(Q_d) = \frac{d(d-1)}{2} + 1$.

Proof. We can show that $t_2(Q_d) \leq \frac{d(d-1)}{2} + 1$ by an argument similar to that in the proof of Theorem 3.1. The only difference is that the first vertex called in level $j > 1$ may call its $d-i$ neighbors in level $j+1$ outside of Q_i and at most $j-1$ neighbors in level $j-1$ of Q_i before it must call a vertex in level $j+1$ of Q_i. Note that it will not call the neighbor in level $j-1$ of Q_i from which it received the message. Summing as before, we get $d-i+1+\sum_{j=2}^{i}(d-i+j-1)$ which is $\frac{d(d-1)}{2}+1$ for $i=d$, so $t_2(Q_d) \leq \frac{d(d-1)}{2} + 1$.

To show that $t_2(Q_d) \geq \frac{d(d-1)}{2}$, we modify the scheme described in the proof of Theorem 3.1. The new scheme is identical except that a vertex does not call the vertex from which it received the message. However, it is important to coordinate some of these calls. In particular, when a vertex u and level i calls v, a level $i-1$ neighbor, this occurs simultaneously with a call from v to u. Otherwise, these vertices can send fewer messages and the time would be decreased. The counting is similar to that in the proof of Theorem 3.1. □

Before giving bounds on $t_1(Q_d)$, we note that the following values are known: $t_1(Q_3) = 4$ (a scheme for the lower bound is given in Figure 1 and the upper bound comes from Theorem 3.3), $t_1(Q_4) = 6$ (a scheme for the lower bound is given in Figure 2 and the upper bound comes from a separate exhaustive search argument which is not included here), $t_1(Q_5) = 8$, and $t_1(Q_6) = 10$ (in both cases, there is a scheme for the lower bound and a separate exhaustive search argument for the upper bound, all of which are omitted).

Theorem 3.3 $\frac{3}{2}d \leq t_1(Q_d) \leq \frac{d(d-1)}{2} + 1$.

Proof. The upper bound follows from Theorem 3.2 and the fact that $t_1(Q_d) \leq t_2(Q_d)$.

To show that $t_1(Q_d) \geq \frac{3}{2}d$, we describe a messy broadcast scheme from originator $(00...0)$. Let Q^{00}, Q^{01}, Q^{10}, and Q^{11}, denote the four $d-2$-dimensional subcubes

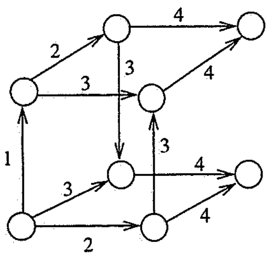

Fig. 1. Messy gossip scheme for Q_3 under model M_1.

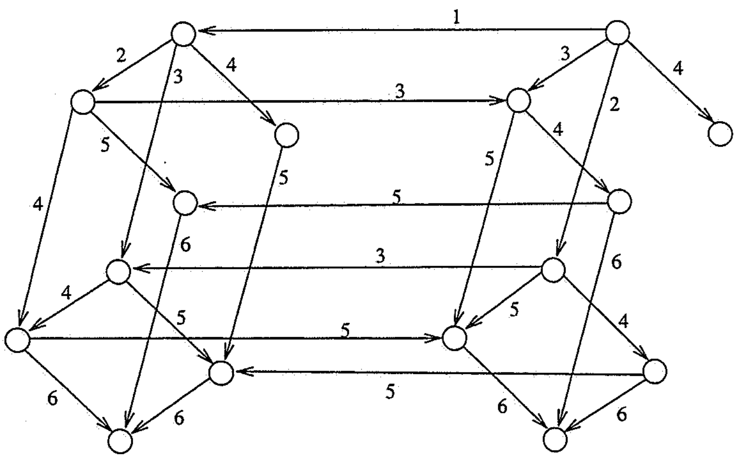

Fig. 2. Messy gossip scheme for Q_4 under model M_1.

with labels of the form (00...), (01...), (10...), and (11...), respectively. The scheme is described recursively based on these subcubes.

Without loss of generality, assume that (00...0) calls (010...0) at time 1. At time 2, these vertices call (10...0) and (0110...0), respectively. At time 3, (00...0) and (0110...0) both call (0010...0) while (010...0) and (10...0) both call (110...0). Thus, only six vertices are informed by time 3. In particular, there are two informed vertices in Q^{00} and Q^{01} and one informed vertex in each of the other subcubes.

Consider the situation in Q^{10} beginning at time 4. Vertex (10...0) is informed and begins messy broadcasting within Q^{10} at time 4 by calling (1010...0). This mimics the call made at time 1 in the original cube where dimension 4 is playing the role of dimension 1 and dimension 3 is playing the role of dimension 2. The two informed vertices (10...0) and (1010...0) call their neighbors (10010...0) and (101010...0), respectively, at time 5. At time 6, (10...0) and (101010...0) both call (100010...0), while (1010...0) and (10010...0) both call (10110...0). Thus, the calls made in this subcube at times 4, 5, and 6 mimic the scheme in the cube at times 1, 2, and 3 with dimensions 4 and 3 playing the roles of dimensions 1 and 2, respectively.

In Q^{00}, the two vertices (00...0) and (0010...0) are both informed by time 3. The vertex (00...0) has already called its neighbor in Q^{10} (at time 2) but (0010...0) has not yet called its neighbor in Q^{10}. In general, a vertex in Q^{00} (other than (00...0)) will begin its messy broadcast after learning the message by first calling its neighbor in Q^{10} and then calling its neighbors within Q^{00}. Thus, at time 4, (00...0) calls (00010...0) and (0010...0) calls (1010...0). This latter call ensures that (1010...0) receives the message from two of its neighbors simultaneously. Vertex (00...0), in fact, will call its neighbor in dimension i at time i for $i \geq 4$. Once vertex (0010...0) calls its neighbor in Q^{10} at time 4, it will continue by calling its neighbor in dimension $i-1$ at time i for all $i \geq 5$. Vertex (00010...0), which is informed at time 4, calls its neighbor in Q^{10} at time 5. Thus, (10010...0) receives the message from two of its neighbors at time 5. Vertex (00010...0) then calls its neighbor in dimension $i-1$ at time i for all $i \geq 6$. In fact, any vertex of Q^{00} which is informed at time $t > 2$, calls its neighbor in Q^{10} at time $t+1$ and then calls its dimension $i-1$ neighbor at time i for $i \geq t+2$. This means that vertices (10110...0) and (100010...0) both receive the message from two neighbors at time 6.

At time 6, the situation in Q^{10} is analogous to the situation in the entire cube at time 3. That is, six vertices are informed, two in each of two $d-4$-dimensional subcubes of Q^{10} and one in the other two $d-4$-dimensional subcubes. The algorithm continues recursively in Q^{10}.

Beginning at time 4, the subcubes Q^{01} and Q^{11} behave similarly, with the broadcast within Q^{01} delayed by 1 time unit immediately after each vertex has been informed.

Under this scheme, it is easy to see that messy broadcasting completes at time $3 + t_1(Q_{d-2})$, thus $t_1(Q_d) \geq t_1(Q_{d-2}) + 3$. For odd d, we get $t_1(Q_d) \geq t_1(Q_5) + 3(\frac{d-5}{2}) > \frac{3}{2}d$. For even d, we obtain $t_1(Q_d) \geq t_1(Q_4) + 3(\frac{d-4}{2}) = \frac{3}{2}d$. □

4. Messy Broadcasting in Complete d-ary Trees

A *full d-ary tree* can be defined recursively as follows: a full d-ary tree is either a single vertex (the root) or it consists of a distinguished vertex (the root) with d children, each of which is the root of a full d-ary tree. A *complete d-ary tree* is a full d-ary tree such that every leaf is at the same distance $(h-1)$ from the root. (The *height* of such a complete d-ary tree is h.) We use $T_{d,h}$ to denote the complete d-ary tree of height h.

Theorem 4.4 $t_3(T_{d,h}) = (d+1)(2h-3)$.

Proof. In $T_{d,h}$, any two leaves which have the root as their lowest common ancestor are at distance $2(h-1)$ apart. The unique path between these two leaves, say u and v, contains the root (which has degree d) and $2h-4$ other intermediate vertices (of degree d). From Lemma 2.3, we get $t_3(u,v) \leq 1+d+(2h-4)(d+1) = (d+1)(2h-3)$. Hence, $t_3(T_{d,h}) \leq (d+1)(2h-3)$.

To show that $t_3(T_{d,h}) \geq (d+1)(2h-3)$, consider a broadcast from the leftmost leaf of $T_{d,h}$ which we will label v_0. Label v_0's parent as v_1 and continue labeling v_i's parent as v_{i+1} until reaching the root (which receives label v_{h-1}). Label v_{h-1}'s rightmost child as v_h and continue down the right side of $T_{d,h}$, labelling v_i's rightmost child as v_{i+1} until reaching the rightmost leaf (which receives label v_{2h-2}).

In a messy broadcast from v_0, at time 1, v_0 informs v_1. At time 2, v_1 sends the message to v_0. At times $3, \ldots\ d+1$, v_1 sends the message to its other children and at time $d+2$, v_1 sends the message to v_2. Continuing in this fashion, at time $id+i+1$, where $1 \leq i \leq h-2$, v_i sends the message to v_{i+1}, so that the message reaches the root (v_{h-1} at time $t = (h-2)d + (h-1)$. The root sends the message back to v_{h-2} at time $t+1$ and then forwards it to its other children at times $t+2, t+3, \ldots, t+d$ such that v_h receives the message at time $t+d$. Vertex v_h sends the message back to v_{h-1} at time $t+d+1$ and then sends it to its children at times $t+d+2, \ldots, t+d+2d+1$ such that v_{h+1} receives the message at time $t+d+2d+1$. This process continues with v_{h+i} receiving the message from v_{h+i-1} at time $t+(i+1)d+i$. Thus, v_{2h-2} receives the message at time $t+(h-1)d+(h-2) = (h-2)d+(h-1)+(h-1)d+(h-2) = (2h-3)(d+1)$. □

When broadcasting in a tree, although there is a subtle distinction between models M_1 and M_2, this distinction will not affect the worst case behavior of a broadcast scheme. In model M_1, each informed vertex must transmit the broadcast message to one of its uninformed neighbors, if any, in each time unit. In model M_2, each informed vertex must transmit the broadcast message to one of its neighbors other than the ones that it knows are informed, if any, in each time unit. At the time a vertex u (other than the originator) is informed by a call from neighbor v, under either model u has exactly one neighbor (v) that it knows to be informed. Thus, u must eventually call all of its other neighbors under either model.

Theorem 4.5 $t_1(T_{d,h}) = t_2(T_{d,h}) = d(2h-3)$.

Proof. As in the proof of Theorem 4.4, consider two leaves u and v at distance $2(h-1)$ apart. From Lemma 2.3, we get $t_1(u,v) \leq t_2(u,v) \leq 1 + d + (2h-4)(d+1) - (2h-3) = d(2h-3)$. Hence, $t_3(T_{d,h}) \leq d(2h-3)$.

As in the proof of Theorem 4.4, we describe a broadcast from the leftmost leaf of $T_{d,h}$ (v_0). In particular, we are interested in the time at which the rightmost leaf (v_{2h-2}) is informed. The scheme (under either model M_1 or M_2 is the same as for M_3 except that no vertex sends the message back to the vertex from which it was received. Thus, the $2h-3$ intermediate vertices each make one fewer call and v_{2h-2} is informed at time $(d+1)(2h-3) - (2h-3) = d(2h-3)$. □

5. Comparing Messy Broadcasting Time with Broadcasting Time in Various Graphs

In Table 1, we summarize the upper bounds obtained in the previous sections for K_n, P_n, C_n, Q_d, and $T_{d,h}$. In each case, we also give the known lower and upper bounds on (regular) broadcasting time in these graphs. (In the case of $T_{d,h}$, the broadcasting time is reported here for the first time, as far as we know.)

Table 1. Comparison of Messy bounds and Simple Broadcast Bounds

graph	M_3 upper bound	M_1 and M_2 upper bound	broadcast lower bound	broadcast upper bound
K_n	$n-1$	$n-1$	$\lceil \log n \rceil$	$\lceil \log n \rceil$
P_n	$2n-3$	$n-1$	$n-1$	$n-1$
C_n	$n-1$	$\lceil \frac{n}{2} \rceil$	$\lceil \frac{n}{2} \rceil$	$\lceil \frac{n}{2} \rceil$
Q_d	$\frac{d(d+1)}{2}$	$\frac{d(d-1)}{2}+1$	d	d
$T_{d,h}$	$(d+1)(2h-3)$	$d(2h-3)$	$(d+1)(h-1)-1$	$(d+1)(h-1)-1$
CCC_d	$3\lfloor \frac{5d}{2} \rfloor-3$	$2\lfloor \frac{5d}{2} \rfloor-1$	$\lceil \frac{5d}{2} \rceil-1$	$\lceil \frac{5d}{2} \rceil-1$
SE_d	$6d-3$	$4d-1$	$2d-1$	$2d-1$
BF_d	$4\lfloor \frac{3d}{2} \rfloor$	$3\lfloor \frac{3d}{2} \rfloor+1$	$1.7417d$	$2d-1$
DB_d	$4d$	$3d+1$	$1.3131d$	$\frac{3}{2}(d+1)$

In addition, we have given the upper bounds obtained from Corollary 2.1 for the cube-connected cycles network of dimension d (CCC_d), the shuffle-exchange network of dimension d (SE_d), the butterfly network of dimension d (BF_d), and the DeBruijn network of dimension d (DB_d). The bounds for broadcasting time in these graphs are found in various papers: [6] for the bounds on CCC_d, [3] for the bounds on SE_d, [4] for the bounds on BF_d, and [4] and [2] for the lower and upper bounds, respectively, on DB_d.

Acknowledgements

We would like to thank Cyrille Micheneau and Sandrine Vial for discussions leading to the results on d-ary trees.

References

[1] R. Ahlswede, H. S. Haroutunian, and L. H. Khachatrian, Messy Broadcasting in Networks, *Communications and Cryptography,* eds. R. E. Blahut, D. J. Costello, Jr., U. Maurer, and T. Mittelholzer (Kluwer, Boston/Dordrecht/London, 1994) 13–24.
[2] J.-C. Bermond and C. Peyrat, Broadcasting in De Bruijn Networks, in *Proc. Nineteenth SE Conf. on Combinatorics, Graph Theory and Computing* (1988) 283–292.
[3] J. Hromkovic, C.-D. Jeschke, and B. Monien, Optimal Algorithms for Dissemination of Information in Some Interconnection Networks, *Algorithmica* **10** (1993) 24–40.
[4] R. Klasing, B. Monien, R. Peine, and E. Stohr, Broadcasting in Butterfly and DeBruijn Networks, *Discr. Appl. Math.* **53** (1994) 183–197.
[5] W. Kndel, New Gossips and Telephones, *Discr. Math.* **13** (1975) 95.
[6] A. L. Liestman and J. G. Peters, Broadcast Networks of Bounded Degree, *SIAM J. Discr. Math.* **1** (1988) 531–540.