Improved Messy Broadcasting in Hypercubes and Simple Graphs *

Thomas E. Hart
Department of Mathematics and Computer Science
Brandon University
Brandon, MB, R7A 6A9
Canada

Hovhannes A. Harutyunyan
Department of Computer Science
Concordia University
Montreal, QC, H3G 1M8
Canada

May 15, 2002

Abstract

In this paper, we present some advances in the study of broadcasting under the *messy broadcasting* model, under which communication between vertices in a communication network is completely uncoordinated. We derive the messy broadcast time of complete bipartite graphs. We also present a proof of the messy broadcast time of hypercube with dimension less than 7 under messy broadcast model M_1, including a non-trivial argument for an upper bound of the 4-dimensional hypercube. In addition, we improve the lower bound on the messy broadcast time of arbitrary-dimensional hypercubes.

1 Introduction and Definitions

Broadcasting is an information dissemination problem in which one node of a communication network must transmit a message to all other nodes in the network; the study of the problem is outlined in [8]. Broadcasting

*This work was supported by the Natural Sciences and Engineering Research Council of Canada

has applications in both computer networks and multiprocessor computer systems. We model the communication network in question as a graph or digraph; the vertices of the graph can represent computers, for example, and the edges can then model communication lines between the machines. We search for graphs whose topologies yield efficient broadcast algorithms.

Formally, let $G = (V, E)$ (or $D = (V, \vec{E})$), and let $u \in V(G)$ (or $V(D)$) be the *originator* of the broadcast. When broadcasting begins, the originator is the only informed vertex. We assume that communication takes place in discrete time units. During each unit of time, each informed vertex *calls* one uninformed adjacent vertex; that vertex is then informed. Vertices may make only one call per time unit, but may receive from any number of vertices in the same time unit. Each call takes exactly one unit of time. Broadcasting is completed when all vertices are informed. The set of all calls made is called the *broadcast protocol* or *broadcast scheme*.

There are various models of broadcasting; each model reflects a different set of assumptions concerning the abilities of the communication network. In the *classical broadcasting* model, every vertex of the graph is assumed to broadcast using an optimal scheme. Research in classical broadcasting focuses on the construction of *broadcast graphs*, in which, for any originator in the graph, there is a broadcast scheme that will finish in $\lceil \log_2 n \rceil$ time units, where n is the number of vertices in the broadcast graph. Since the number of informed vertices can, at most, double during each time unit, this is the minimum possible time. This model assumes that it is possible to provide all vertices in a communication network with enough information to ensure optimal communication.

In this paper, however, we study a model called *messy broadcasting*, which assumes that we are unable to provide the vertices with any significant amount of information. In this type of broadcasting, every vertex acts independently, without knowing which vertex is the originator of the message, or the time at which the message was sent; futhermore, besides knowing which vertices are its neighbours, each vertex has no knowledge of the network's topology. All informed vertices thus broadcast to randomly-chosen neighbours at each time unit. This variant of broadcasting was introduced by Ahlswede, Haroutunian, and Khatchatrian in [1], in which they introduced the following three models for messy broadcasting:

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

Model M_2: Every informed vertex knows from which vertex or vertices it received the broadcast message, and to which neighbours 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 neighbours other than the ones that it knows are informed, if any exist, in each time unit.

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

When studying messy broadcasting, we are concerned with the worst-case performance of a given graph G. The *broadcast time of vertex* u in a graph G under model M_i, denoted t_i, $1 \leq i \leq 3$, is defined to be the maximum number of time units required to inform all vertices of G, with u as the originator. The definition of the *broadcast time of a graph* G as $t_i(G) = max\{t_i(u) \mid u \in V\}$ is then natural.

When studying messy broadcasting, our job becomes the search for graphs whose topologies prohibit grossly inefficient broadcast schemes. In addition, we attempt to derive the messy broadcast times of graphs that are good for classical broadcasting, in order to analyze the effect of the messy broadcasting model on communication efficiency.

Messy broadcasting seems promising. It is technically easy to build a cheap and reliable network for messy broadcasting. In addition, it is already shown [1] that, for some common communication networks, messy broadcasting is almost as fast as classical broadcasting. When n is large, there are graphs G and H on n vertices such that $t_1(G) \leq t_2(G) \leq 1.89 \log_2 n$, and $t_3(H) \leq 2.5 \log_2 n$.

2 Messy Broadcasting in Complete Bipartite Graphs

In [5], Harutyunyan and Liestman proved that $t_i(K_n) = n-1$, for $i = 1, 2, 3$. We can apply similar techniques to complete bipartite graphs.

Let $K_{m,n}$ denote the complete bipartite graph with $V(K_{m,n}) = A \cup B$, where $A = \{u_1, u_2, \ldots, u_m\}$, $B = \{v_1, v_2, \ldots, v_n\}$, $A \cap B = \emptyset$, and $E(K_{m,n}) = \{\, u_i v_j \mid 1 \leq i \leq m,\ 1 \leq j \leq n\}$; as an example, $K_{3,2}$ is pictured in Figure 1. Then,

Lemma 2.1 $t_1(K_{m,n}) = t_2(K_{m,n}) = max\{m, n\}$.

Proof. Without loss of generality, we will assume that the originator is u_1, and that it calls v_1 at time $t = 1$; hence, after time 1, u_1 and v_1 are the only informed vertices. Beginning at time $t = 2$, u_1 must call its $n - 1$

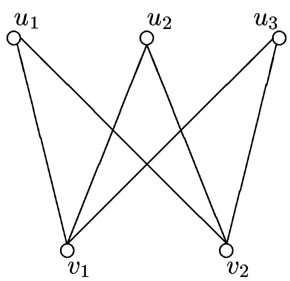

Figure 1: $K_{3,2}$, where $A = \{u_1, u_2, u_3\}$ and $B = \{v_1, v_2\}$

uninformed neighbours, and v_1 must call its $m-1$ uninformed neighbours, ensuring that broadcasting completes no later than at time $max\{m, n\}$.

To show that it is possible to take $max\{m, n\}$ time units to complete broadcasting, consider the following scheme: at time i, all informed vertices in A broadcast to v_i if $i \leq n$, and all informed vertices in B broadcast to u_i if $i \leq m$. This completes our proof. $\Box$

Lemma 2.2 $t_3(K_{m,n}) = max\{m, n\} + 1$.

Proof. We must consider two distinct cases:

Case 1: Again without loss of generality, we will assume that the originator is u_1, and that it calls v_1 at time $t = 1$; hence, after time 1, u_1 and v_1 are the only informed vertices. Beginning at time $t = 2$, u_1 must call its $n-1$ uninformed neighbours, and v_1 must call its m neighbours, ensuring that broadcasting completes no later than at time $max\{m+1, n\}$.

To show that it is possible to take $max\{m+1, n\}$ time units to complete broadcasting, consider the following scheme: at time i, all informed vertices in A broadcast to v_i if $i \leq n$, and all informed vertices in B broadcast to u_{i-1} if $i \leq m+1$.

Case 2: Assume that v_1 instead of u_1 is the originator. Then, by a similar argument, broadcasting completes in $max\{m, n+1\}$ time units.

Since, depending on whether the originator is u_1 or v_1, messy broadcasting completes in either $max\{m+1, n\}$ or $max\{m, n+1\}$ time units, respectively, $t_3(K_{m,n}) = max\{m, n\} + 1$, and our proof is complete. $\Box$

A simple topology often used in real networks is that of a cycle; it is used, for example, in some LANs. In [5], it was shown that $t_1(C_n) =$

$t_2(C_n) = \lceil \frac{n}{2} \rceil$, and that $t_1(C_n) = n - 1$. By choosing m and n so that they differ by at most 1, we can construct a bipartite graph $K_{m,n}$ such that $t_1(K_{m,n}) = t_2(K_{m,n}) = \lceil \frac{m+n}{2} \rceil$ and $t_3(K_{m,n}) = \lceil \frac{m+n}{2} \rceil + 1$. This makes complete bipartite graphs, in the worst-case, as fast as cycles under two of our models, and faster than cycles under the third. In addition, the average-case time of complete bipartite graphs must be lower than that of cycles, since messy broadcasting in C_n can be completed in no fewer than $\lceil \frac{n}{2} \rceil$ time units, while messy broadcasting in $K_{m,n}$ can be completed much more quickly. The cost, of course, is that a complete bipartite graph has more edges than a cycle with the same number of vertices. This relatively fast messy broadcast time, along with the simplicity of the complete bipartite graph, may make it a promising topology for connecting a small number of machines.

3 Exact Values of $t_1(Q_k)$ for $1 \leq k \leq 6$ Proved

The k-dimensional hypercube Q_k is defined as follows: $Q_0 = K_1$, $Q_1 = K_2$, and $Q_k = Q_{k-1} \times Q_1$ for $k \geq 2$. Slightly less formally, we can say that we construct a k-dimensional hypercube by taking two $(k-1)$-dimensional hypercubes, and connecting each vertex with its counterpart in the other hypercube. Hypercubes are of interest in the study of broadcasting because they were the first infinite class of *minimum broadcast graphs* to be discovered, and because they are often used as the topology of real-life networks.

Vertices in Q_k are typically labelled with k-bit binary strings. Without loss of generality, the originator in a messy broadcast scheme is assumed to be vertex $(00\ldots0)$, and it is usually assumed to call $(10\ldots0)$ at time 1, $(01\ldots0)$ at time 2, $\ldots$, and $(00\ldots01)$ at time k.

Exact values for $t_2(Q_k)$ and $t_3(Q_k)$ were found by Harutyunyan and Liestman in [5]. In addition, the exact value of $t_1(Q_k)$ is were found for $1 \leq k \leq 6$. These values are summarized in the following table:

$\mathbf{k}$	$\mathbf{t_1(Q_k)}$
1	1
2	2
3	4
4	6
5	8
6	10

For $k \geq 4$, however, the proofs of these statements were omitted. The upper bounds follow from more difficult exhaustive searches, all of which were omitted in [5]. Here, we present a proof that $t_1(Q_4) = 6$; the proofs for Q_5 and Q_6 are similar, but more grueling.

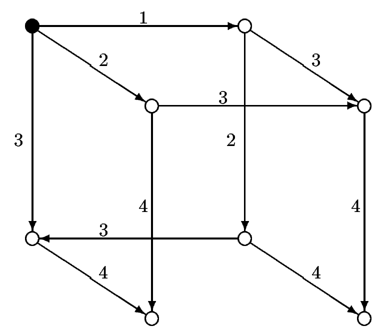

Figure 2: Time 4 messy broadcast scheme for Q_3

Observing the above table, we can see that $t_1(Q_k) = 2k-2$ for $2 \leq k \leq 6$. It should be noted that this is not true in general, since we know that $t_1(Q_{10}) \geq 19 > 18 = 2(10) - 2$. We know that $t_1(Q_{10}) \geq 19$ because a computer was able to generate a messy broadcast scheme for Q_{10} taking 19 time units, using an unsophisticated search algorithm.

Lemma 3.1 $t_1(Q_4) = 6$.

Proof: The fact that $t_1(Q_4) \geq 6$ is proved by the existence of a messy broadcast scheme for Q_4 taking 6 time units, as shown in Figure 3.

Proving that $t_1(Q_4) \leq 6$ is more difficult. Assume that vertex (0000) is the originator of a messy broadcast. We shall show that all vertices in Q_4 must be informed by time 6. Recall that the vertices of Q_4 are labelled (0000) through (1111), and that a vertex v is on level i of Q_4 iff there are i 1's in the label of v; for example, vertex (1001) is on level 2. Without loss of generality, we shall assume that vertex (0000) broadcasts first to (1000), then (0100), (0010), and (0001). Then all vertices on level 1 of Q_4 are informed at or before time 4.

Since vertex (1000) is informed at time 1 and must call its neighbours in some order, vertices (1100), (1010), and (1001) must all be informed by time 4. Similarly, vertex (0100) is informed at time 2, and must call (1100), (0110), and (0101) in some order, so these vertices must all be informed by time 5. This leaves vertex (0011). Vertex (0010) is informed at time 3, and must call its three neighbours on level 2, one of which is (0011), in the following time units. Hence all vertices on level 2 of Q_4 must be informed by time 6.

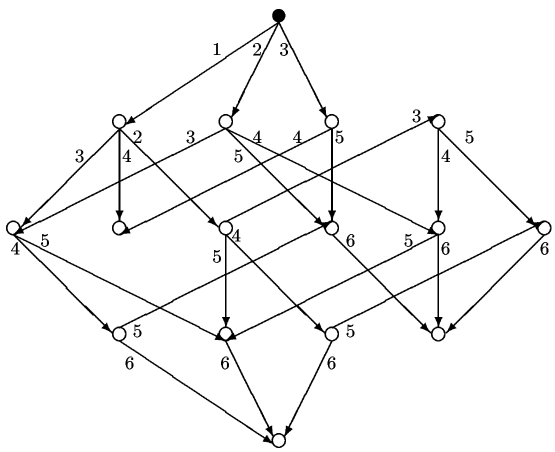

Figure 3: Time 6 messy broadcast scheme for Q_4

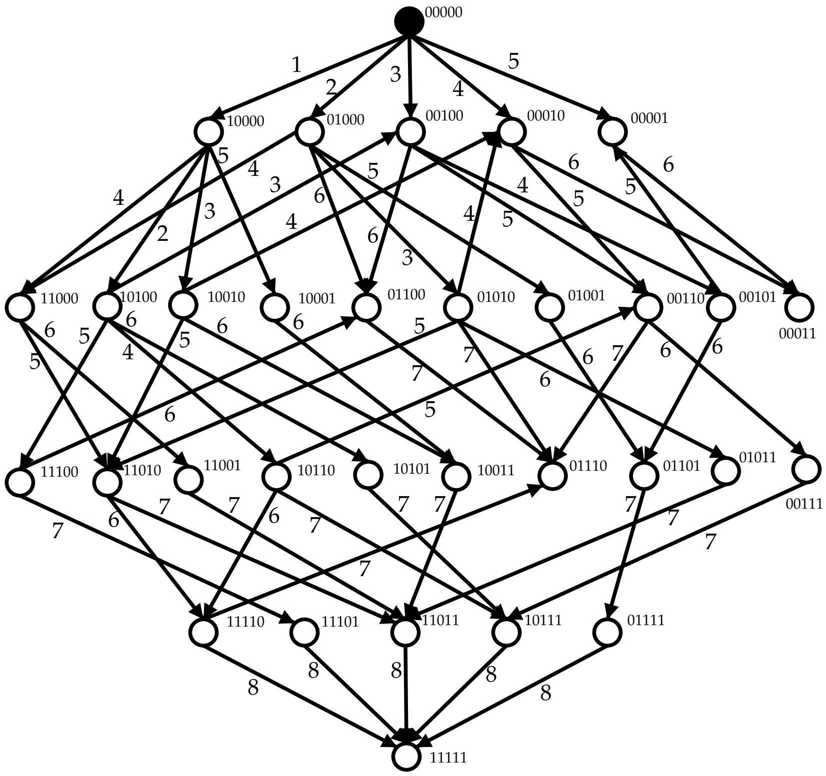

Figure 4: Time 8 messy broadcast scheme for Q_5

As mentioned above, vertex (1000) will inform vertices (1100), (1010), and (1001) by time 4. Each of the three vertices (1110), (1101), and (1011) at level 3 is connected to exactly two of these vertices. It follows that all three of these level 3 vertices must be informed by time 6. This leaves vertex (0111). Recall that (0110) and (0101) are neighbours of both (0100) and (0111), and that (0100) is informed at time 2. Thus one of (0110) or (0101) must be informed by (0100) by time 4; call this vertex u. Since all vertices at level 1 are informed at time 4, and u has two neighbours at each of levels 1 and 3, u must broadcast to its neighbours at level 3 at times 5 and 6. This ensures that (0111) is informed by time 6; that is, all vertices on level 3 of Q_4 must be informed by time 6.

It remains to be shown only that vertex (1111) must be informed by time 6. If any vertex on level 3 of Q_k is informed at time 3, then (1111) must be informed by time 6, since any vertex on level 3 is a neighbour of (1111), and may have up to three uninformed neighbours, which it must call during times 4, 5, and 6. Similarly, if a vertex on level 3 is informed at time 4, and at least two of its neighbours on level 2 are also informed by time 4, then vertex (1111) must be informed by time 6. We shall use these two facts to prove that (1111) must be informed by time 6. Let us consider three cases.

If vertex (1000) broadcasts to (1100) at time 2, then (1100) must broadcast to a vertex at level 3 at time 3, since its only other neighbour at level 1, vertex (0100), is also informed at time 2. Then (1111) must be informed by time 6.

Say vertex (1000) broadcasts to (1010) at time 2. Then vertex (1010) can broadcast to (0010) at time 3, after which it must broadcast to vertices at level 3. If (1010) broadcasts to (1110) at time 4, then (1111) will be informed by time 6, since one of (1110)'s neighbours at level 2, (1100), must be informed by time 4. Similarly, if (1010) broadcasts to (1011) at time 4, then (1111) will be informed by time 6, since one of (1011)'s neighbours at level 2, (1001), must be informed by time 4. In either case, (1111) must be informed by time 6.

Say vertex (1000) broadcasts to (1001) at time 2. If (1001) broadcasts to a vertex at level 3 at time 3, then (1111) will be informed by time 6. So let us assume that (1001) broadcasts to (0001) at time 3. Then (1001) must broadcast to either (1011) or (1101) at time 4. But (1011) is a neighbour of (1010), which must be informed by time 4 since it is a neighbour of (1000). Similarly, (1101) has neighbour (1100), which also must be informed by time 4. In either case, a vertex on level 3 is informed at time 4 and has two of its neighbours on level 3 informed by time 4, so vertex (1111) must be informed by time 6.

Hence, all vertices in Q_4 must be informed by time 6. This completes our proof. □

Combined with the trivial proofs that $t_1(Q_2) = 2$ and $t_1(Q_3) = 4$, and the non-trivial arguments for Q_5 and Q_6 which are similar to the argument for Q_4, above, we have:

Theorem 3.1 *$t_1(Q_k) = 2k - 2$ for $2 \leq k \leq 6$.*

Proof. The lower bounds are arrived at relatively easily by constructing messy broadcast schemes. A broadcast scheme in Q_2 taking two time units is trivially constructed. A messy broadcast scheme for Q_3 taking 4 time units is shown in Figure 2, and a messy broadcast scheme for Q_4 is shown in Figure 3. Figure 4 shows a messy broadcast scheme for Q_5 taking 8 time units; since this diagram is complex, we include the binary labels of the vertices to make the diagram readable. We omit a messy broadcast scheme for Q_6, since such a diagram would be too large and complex to be instructive.

As mentioned above, the upper bounds for Q_2 and Q_3 are trivial, the upper bound for Q_4 is included above, and the upper bounds for Q_5 and Q_6 are proved in a similar fashion to the upper bound for Q_4. Hence, we have our result. $\square$

We should re-iterate that it is known that $t_1(Q_k) \neq 2k - 2$ in general; however, the smallest k such that $t_1(Q_k) > 2k - 2$ is unknown, although we know that it is less than or equal to 10.

4 An Improved Lower Bound on $t_1(Q_k)$

As mentioned above, exact values for $t_2(Q_k)$ and $t_3(Q_k)$ were found in [5]. For model M_1, however, the best bound found was $\frac{3}{2}k \leq t_1(Q_k) \leq \frac{k(k-1)}{2} + 1$. We were able to obtain improved lower bounds for $t_1(Q_k)$ through two successive generalizations of Harutyunyan and Liestman's proof from [5].

Theorem 4.1 $\frac{5k-2}{3} \leq t_1(Q_k) \leq \frac{k(k-1)}{2} + 1$.

Proof. The upper bound is $t_2(Q_k)$, which was found in [5].

To show that $t_1(Q_k) \geq \frac{5k-2}{3}$, we describe a messy broadcast scheme from originator $(000\ldots0)$. We begin by partitioning our hypercube into eight distinct sub-cubes: Q^{000}, Q^{100}, Q^{010}, Q^{001}, Q^{110}, Q^{101}, Q^{011}, and Q^{111}. In general, the leftmost three bits of the binary labels of all vertices in Q^{xyz} are xyz. For $t = 1, 2, \ldots, 5$, we follow the scheme shown in Figure 5.

Consider the sub-cubes Q^{100} and Q^{110}. Beginning at time 6, Q^{110} broadcasts internally according to the worst-case messy broadcast scheme for Q_{k-3}. All neighbours of $(100\ldots00)$ in Q^{100} broadcast to their neighbours in Q^{110}, and then proceed to make calls within Q^{100} according to the worst-case messy broadcast scheme for Q_{k-3}. Calls in Q^{100} to vertices with a lower level are then made in sync with the corresponding calls in Q^{110}; this is illustrated in Figure 6. Analogous situations occur with the pairs of sub-cubes Q^{101} and Q^{111}, Q^{010} and Q^{110}, and Q^{101} and Q^{111}; hence, these six sub-cubes finish broadcasting simultaneously.

A similar situation occurs with the sub-cube pairs Q^{000} and Q^{010}, and Q^{001} and Q^{011}. Neighbours of $(000\ldots00)$ other than $(0001\ldots00)$ broadcast to their neighbours in Q^{010}, and then continue with their internal broadcast scheme (similarly for the other pair of sub-cubes); hence, these four sub-cubes complete broadcasting simultaneously. Note that, since internal broadcasts from $(0001\ldots00)$ have already been delayed by its call to $(0011\ldots00)$, the fact that it does not broadcast to its neighbour in Q^{010} will not cause problems. Furthermore, since two of these four sub-cubes have been shown to finish broadcasting at the same time as the other four sub-cubes of Q_k, we can see that all eight sub-cubes complete broadcasting simultaneously.

Since Q^{110} and Q^{111} complete broadcasting in $t_1(Q_{k-3})$, and the set-up for this scheme requires 5 time units, we can see that broadcasting completes at time $5+t_1(Q_{k-3})$; hence, $t_1(Q_k) \geq t_1(Q_{k-3})+5$, and $t_1(Q_k) \geq t_1(Q_{k-3i}) + 5i$, when $i \leq \frac{k}{3}$.

Solving the latter recurrence relation for the case where $k \bmod 3 = 1$, we get $t_1(Q_k) \geq t_1(Q_4) + 5(\frac{k-4}{3}) = \frac{5}{3}k - \frac{2}{3}$. For $k \bmod 3 = 2$, we get $t_1(Q_k) \geq t_1(Q_5) + 5(\frac{k-5}{3}) = \frac{5}{3}k - \frac{1}{3}$. And for $k \bmod 3 = 0$, we get $t_1(Q_k) \geq t_1(Q_6) + 5(\frac{k-6}{3}) = \frac{5}{3}k$. In all three cases, the result is greater than or equal to our lower bound of $\frac{5k-2}{3}$; hence, our theorem is proved. □

In a similar, but slightly more complex manner, we obtained,

Theorem 4.2 $\frac{7k-5}{4} \leq t_1(Q_k) \leq \frac{k(k-1)}{2} + 1$.

Proof. This result is a generalization of Theorem 4.1. From Figure 7, we can see that $t_1(Q_k) \geq t_1(Q_{k-4}) + 7$. When $t \geq 8$, broadcasting occurs in the same fashion as shown in Figure 6, as is true for Theorem 4.1.

Since $t_1(Q_k) \geq t_1(Q_{k-4}) + 7$, $t_1(Q_k) \geq t_1(Q_{k-4i}) + 7i$, when $i \leq \frac{k}{4}$. For the case in which $k \bmod 4 = 0$, we get that $t_1(Q_k) \geq t_1(Q_4) + 7(\frac{k-4}{4}) = \frac{7k-4}{4}$. When $k \bmod 4 = 1$, we get that $t_1(Q_k) \geq t_1(Q_5) + 7(\frac{k-5}{4}) = \frac{7k-3}{4}$. When $k \bmod 4 = 2$, we get that $t_1(Q_k) \geq t_1(Q_6) + 7(\frac{k-6}{4}) = \frac{7k-2}{4}$. When $k \bmod 4 = 0$, we get that $t_1(Q_k) \geq t_1(Q_3) + 7(\frac{k-3}{4}) = \frac{7k-5}{4}$. In all four cases, $t_1(Q_k) \geq \frac{7k-5}{4}$. This completes our proof. □

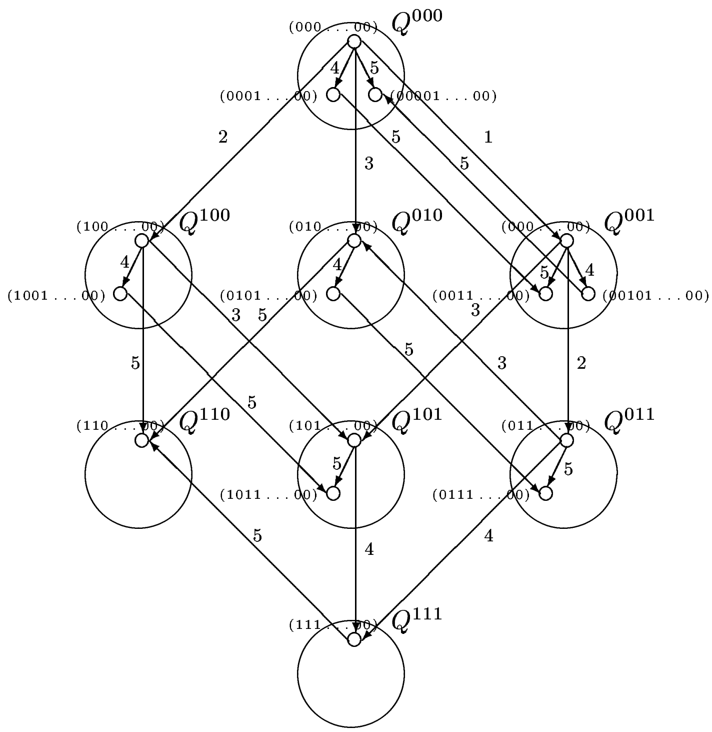

Figure 5: Setup Scheme for Theorem 4.1

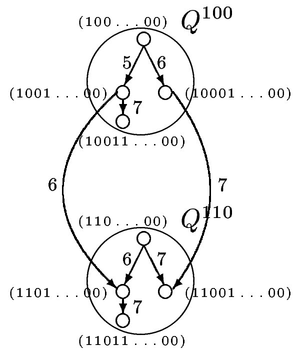

Figure 6: Broadcasting in Q_{100} and Q_{110} after time $t = 7$ for Theorem 4.1; note the synchronized calls

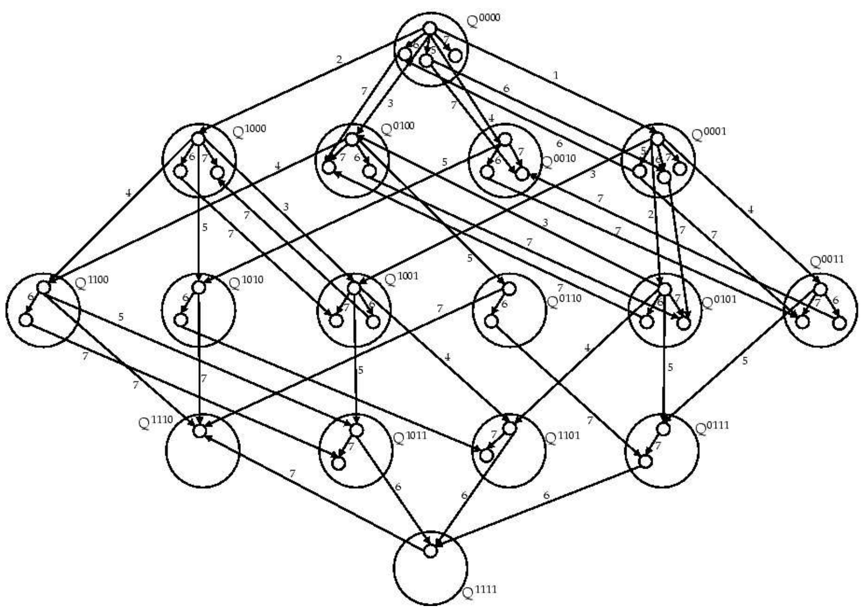

Figure 7: Setup scheme for Theorem 4.2

5 A Note on Messy Broadcasting in Hypercubes Under Model M_1

To better understand messy broadcasting in hypercubes under model M_1, we simulated the process using a computer. We first tried broadcasting completely randomly, then using simple techniques to produce collisions.

The most theoretically-interesting result of these simulations is that our bounds on $t_1(Q_k)$ arrived at in Section 4 fall significantly short of the true value of $t_1(Q_k)$. For example, our simulations produced a broadcast scheme for Q_{22} taking 53 time units. This makes it extremely unlikely that $t_1(Q_k)$ is asymptotically linear; however, no method to produce a non-linear lower bound on $t_1(Q_k)$ has been found.

Another result of some interest is that the mean time taken to complete messy broadcasting in Q_k seems to increase linearly with k. To use the same example as above, messy broadcasting in Q_{22} completed, on average, in 31 time units. A similar situation occurs in K_n, where $t_1(K_n) = t_2(K_n) = t_3(K_n) = n - 1$, but we can expect messy broadcasting in K_n to complete much more quickly the vast majority of the time. This suggests that the worst-case messy broadcast times examined thus far may not be the only appropriate metric by which to examine and compare the performance of different graphs under the messy broadcasting model.

6 Conclusions

We continued our study of messy broadcasting, in which all broadcasts in a given network are made randomly. Our finding of the messy broadcast times of $K_{m,n}$ applies the study of messy broadcasting to a simple topology. This result gives a simple interconnection scheme that is as fast as a cycle topology under models M_1 and M_2, and faster than a cycle under model M_3. This gives us a simple way to connect a small number of machines relatively efficiently.

Our proof that $t_1(Q_k) = 6$ illustrates the difficulty of establishing upper bounds on the messy broadcast times of graphs that are not entirely simple. Model M_1 may suffer especially from this problem for graphs whose vertices have high degrees, such as hypercubes, since the set of vertices to which a vertex may broadcast is affected by the actions vertices other than itself and its neighbours. This key difference may make the performance of model M_1 significantly more difficult to analyze in complicated graphs.

Despite the difficulties in working with model M_1, we improved the lower bound on $t_1(Q_k)$ by generalizing previous results. Our improved lower bound, however, falls short of the still-unknown value of $t_1(Q_k)$. The problem of how to arrive at a non-linear lower bound for $t_1(Q_k)$ remains unsolved.

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. Mauter, and T. Mittelholzer (Kluwer, Boston/Dordrecht/London, 1994) 13-24.

[2] J.-C. Bermond and P. Fraigniaud, Broadcasting and Gossiping in de Bruijn Networks, *SIAM J. Comput.*, Vol. 23, No. 1, pp. 212-225, 1994.

[3] P. Fraigniaud and E. Lazard, Methods and problems of communication in usual networks, *Discrete Applied Mathematics* 53, pp. 79-133, 1994.

[4] L. Gargano, A. Pelc, S. Perennes, and U. Vaccaro, Efficient Communication in Unknown Networks, 26th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2000) Konstanz, Germany, June 15-17, 2000.

[5] H. A. Harutyunyan and A. L. Liestman, Messy Broadcasting, *Parallel Processing Letters* Vol. 8 No. 2, pp. 149-159, 1998.

[6] H. A. Harutyunyan and A. L. Liestman, More Broadcast Graphs, *Discrete Applied Mathematics* 98, pp.81-102, 1999.

[7] H. A. Harutyunyan and T. E. Hart, Messy Broadcasting in Common Interconnection Networks, *Concordia University*, 2002.

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