Gossip versus Deterministic Flooding: Low Message Overhead and High Reliability for Broadcasting on Small Networks

Meng-Jang Lin
Department of Electrical and Computer Engineering
The University of Texas at Austin

Keith Marzullo
Department of Computer Science and Engineering
University of California, San Diego

Stefano Masini
Department of Computer Science
University of Bologna, Italy

Abstract

Rumor mongering (also known as *gossip*) is an epidemiological protocol that implements broadcasting with a reliability that can be very high. Rumor mongering is attractive because it is generic, scalable, adapts well to failures and recoveries, and has a reliability that gracefully degrades with the number of failures in a run. In this paper we present a protocol that superficially resembles rumor mongering but is deterministic. We show that this new protocol has most of the same attractions as rumor mongering. The one attraction that rumor mongering has—namely graceful degradation—comes at a high cost in terms of the number of messages sent. We compare the two approaches both at an abstract level and in terms of how they perform in an Ethernet.

1 Introduction

Consider the problem of designing a protocol that broadcasts messages to all of the processors in a network. One can be interested in different metrics of a broadcast protocol, such as the number of messages it generates, the time needed for the broadcast to complete, or the reliability of the protocol (where *reliability* is the probability that either all or no nonfaulty processors deliver a broadcast message and that all nonfaulty processors deliver the message if the sender is nonfaulty). With the metrics of interest in mind, there are two general approaches one might approach designing a broadcast protocol.

One approach would be to build upon the specific physical properties of the network. For example, there are several broadcast protocols that attain very high reliability for Ethernet networks [16] or redundant Ethernets [2, 5, 7]. Such a protocol can be very efficient in terms of the chosen metrics because one can leverage off of the particularities of the network. On the other hand, such a protocol is not very portable, since it depends so much on the physical properties of the network.

The other approach is to assume a generic network. With this approach, one chooses a set of basic network communication abstractions such as sending and receiving a message. If reliability is a concern, then one can adopt a failure model that is generic enough to apply to many different physical networks. There are many examples of reliable broadcast protocols for such generic networks [24]. We consider in this paper broadcast protocols for generic networks.

Unfortunately, many reliable broadcast protocols do not scale well to large numbers of processors [4]. A family of protocols for generic networks that are scalable are called *epidemiological*

algorithms or *gossip protocols*. Gossip protocols are probabilistic in nature: a processor chooses its *partner* processors with which to communicate randomly. They are scalable because each processor sends only a fixed number of messages, independent of the number of processors in the network. In addition, a processor does not wait for acknowledgments nor does it take some recovery action should an acknowledgment not arrive. They achieve fault-tolerance because a processor receives copies of a message from different processors. No processor has a specific role to play, and so a failed processor will not prevent other processors from continuing sending messages. Hence, there is no need for failure detection or specific recovery actions.

A drawback of gossip protocols is the number of messages that they send. Indeed, one class of gossip protocols (called *anti-entropy* protocols [9]) send an unbounded number of messages in nonterminating runs. Such protocols seem to be the only practical way that one can implement a gossip protocol that attains a high reliability in an environment in which links can fail for long periods of time [23]. Hence, when gossiping in a large wide-area network, anti-entropy protocols are often used to ensure high reliability. However, for applications that require timely delivery, the notion of reliability provided by anti-entropy may not be strong enough since it is based on the premise of *eventual delivery* of messages.

Another class of gossip protocols is called *rumor mongering* [9]. Unlike anti-entropy, these protocols terminate and so the number of messages that are sent is bounded. The reliability may not be as high as anti-entropy, but one can trade off the number of messages sent with reliability. Rumor mongering by itself is not appropriate for networks that can partition with the prolonged failure of a few links, and so is best applied to local-area networks and small wide-area networks.

In this paper, we describe a reliable broadcast protocol for systems in which links can drop messages and in which processors can crash. Our protocol has most of the attractive properties of rumor mongering. However, it usually sends many fewer messages than gossip does to attain the same reliability. Unlike gossip, the protocol is deterministic. It can be thought of as a gossip protocol in which a processor deterministically selects its partners in a way to minimize the number of messages sent while still attaining a desired target reliability. The protocol effectively superimposes a *communications graph* on top of the processors and sends messages only along the edges of this graph. The imposed graph has a minimal number of links while still having a high enough connectivity to attain the desired reliability. We call these graphs *Harary graphs* because the construction we use comes from a paper by Frank Harary [14].

The rest of the paper proceeds as follows. We first discuss some related research. We then describe gossip protocols and their properties. Next, we describe Harary graphs and show how to construct them. We then compare Harary graph-based flooding with gossip protocols and conclude the paper.

2 Related Work

Superimposing a communications graph is a well-known technique for implementing broadcast protocols. Let an undirected graph $G = (V, E)$ represent such a superimposed graph, where each node in V is a processor and each edge in E means that the two nodes incident on the edge can directly send each other messages at the transport level. Two nodes that have an edge between them are called *neighbors*. A simple broadcast protocol has a processor initiate the broadcast of m by sending m to all if its neighbors. Similarly, a node that receives m for the first time sends m to all of its neighbors except for the one which forwarded it m. This technique is commonly called *flooding* [1]. Depending on the superimposed graph structure, a node may be sent more than one copy of m. We call the number of messages sent in the reliable broadcast of a single m the *message*

overhead of the broadcast protocol. For flooding, the message overhead is between one and two times the number of edges in the superimposed graph.

The most common graph that is superimposed is a spanning tree (for example, [10, 21]). Spanning trees are attractive when one wishes to minimize the number of messages: in failure-free runs, each processor receives exactly one message per broadcast and so the message overhead is $|V| - 1$. Their drawback is that when failures occur, a new spanning tree needs to be computed and disseminated to the surviving processors. This is because a tree can be disconnected by the removal of any internal (*i.e.* nonleaf) node or any link.

If a graph more richly connected than a tree is superimposed, then not all sets of link and internal node failures will disconnect the graph. Hence, if a detected and persistent failure occurs, any reconfiguration — that is, the computation of a new superimposing graph — can be done while the original superimposed graph is still used to flood messages. Doing so lessens the impact of the failure.

One example of the use of a graph more richly connected than a tree is discussed in [11]. In this work, they show how a hypercube graph can be used instead of a tree to disseminate information for purposes of garbage collection. It turns out that a hypercube is a Harary graph that is three-connected.

A more theoretical example of the use of a more richly connected graph than a tree is given by Liestman [17]. The problem being addressed in this work is, in some ways, similar to the problem we address. Like our work, they are interested in fault-tolerant broadcasting. And, like us, they wish to have a low message overhead. The models, however, are very different. They consider only link failures while we consider both link and node failures. They assume that a fixed unit of time elapses between a message being sent and it being delivered. They are concerned with attaining a minimum broadcast delivery time while we are not. And, the graphs that they superimpose are much more complex to generate as compared to Harary graphs. However, it turns out that some of the graphs they construct are also Harary graph.

The utility of the graphs described by Harary in [14] for the purposes of reducing the probability of disconnection was originally examined in [22]. This work, however, was concerned with rules for laying out wide-area networks and is unrelated to the work described in this paper.

3 Gossip Protocols

Gossip protocols, which were first developed for replicated database consistency management in the Xerox Corporate Internet [9], have been built to implement not only reliable multicast [4, 12] but also failure detection [20] and garbage collection [13]. Nearly all gossip protocols have been developed assuming the failure model of processor crashes and message links failures that lead to dropped messages. They are not usually discussed in the context of synchronous versus asynchronous or timed asynchronous models [8, 24]. Like earlier work in gossip protocols, we do not consider the question of how synchronous the environment must be to ensure that these protocols terminate in all runs.

Gossip protocols have the following features:

- *scalability* Their performance does not rapidly degrade as the number of processors grow. Each processor sends a fixed number of messages that is independent of the number of processors. And, each processor runs a simple algorithm that is based on slowly-changing information. In the protocol above, a processor needs to know the identity of the other processors on its (local-area or small wide-area) network and a few constants. Hence, as long as the stability

of the physical network does not degrade as the number of processors grow, then gossip is scalable.

- *adaptability* It is not hard to add or remove processors in a network. In both cases, it can be done using the gossip protocol itself.

- *graceful degradation* For many reliable broadcast protocols, there is a value f such that if there are no more than f failures (as defined by a failure model) then the protocol will function correctly. If there are $f + 1$ failures, however, then the protocol may not function correctly. The reliability of the protocol is then equal to the probability that no more than f failures occur. Computing this probability, however, may be hard to do and the computation may be based on values that are hard to measure. Hence, it is advantageous to have a protocol whose probability of functioning correctly does not drop rapidly as the number of failures increases past f. Such a protocol is said to *degrade gracefully*. One can build gossip protocols whose reliability is rather insensitive to f.

There are many variations of gossip protocols within the two approaches mentioned in Section 1. Below is one variation of rumor mongering. This is the protocol that is used in [15] (specifically, $F = 1$ and the number of hops a gossip message can travel is fixed) and is called *blind counter rumor mongering* in [9]:

```
initiate broadcast of m:
   send m to B_initial neighbors

when (p receives a message m from q)
   if (p has received m no more than F times)
      send m to B randomly chosen neighbors that
        p does not know have already seen m;
```

A gossip protocol that runs over a local-area network or a small wide-area network effectively assumes that the communications graph is a clique. Therefore, the neighbors of a processor would be the other processors in the network. A processor knows that another processor q has already received m if it has previously received m from q. Processors can more accurately determine whether a processor has already received m by having m carry a set of processor identifiers. A processor adds its own identifier to the set of identifiers before forwarding m, and the union of the sets it has received on copies of m identify the processors that it knows have already received m. Henceforth in this paper, the gossip protocol we discuss is this specific version of rumor mongering that uses this more accurate method.

Since a processor selects it partners randomly, there is some chance that a message may not reach all processors even when there are no failures. However, in a clique, such probability is small [19]. Therefore, the reliability of gossip protocols is considered to be high. This is not the case when the connectivity of the network is not uniform, though. It has been shown that when the network has a hierarchical structure, a gossip message can fail to spread outside a group of processors [18].

It is often very difficult to obtain an analytical expression to describe the behavior of a gossip protocol. Often, the best one can get are equations that describe asymptotic behavior for simple network topologies. For more complex topologies or protocols, one almost always resorts to simulations. Below we give reliability results obtained from simulations when failures are considered.

B	F	$B \cdot F$	$f = 0$	$f = 1$	$f = 2$	$f = 3$	$f = 4$
2	1	2	0.0290	0.0243	0.0171	0.0150	0.0157
3	1	3	0.3598	0.3242	0.3021	0.2708	0.2547
4	1	4	0.7351	0.7120	0.6781	0.6483	0.6174
2	2	4	0.8011	0.7711	0.7349	0.6880	0.6429
3	2	6	0.9786	0.9724	0.9663	0.9614	0.9458
2	3	6	0.9912	0.9872	0.9844	0.9784	0.9703
4	2	8	0.9982	0.9963	0.9964	0.9956	0.9906
2	4	8	0.9996	0.9996	0.9996	0.9995	0.9987
3	3	9	0.9996	0.9997	0.9997	0.9991	0.9990
4	3	12	1.0000	1.0000	1.0000	0.9999	1.0000
3	4	12	0.9999	0.9999	1.0000	1.0000	0.9999
4	4	16	1.0000	1.0000	1.0000	1.0000	1.0000

Table 1: Measured reliability with 10^4 broadcasts. $n = 32$, $B_{initial} = \min(B, f + 1)$.

Table 1 shows the measured reliability of 10,000 broadcasts among $n = 32$ processors for different values of B, F and f, where f is the number of crashed processors. No link failures occurred during these broadcasts and all processors that crashed did so before the first broadcast. As can be seen, the reliability does not drop too much between $f = 0$ and $f = 4$ since a processor can receive messages from many processors. Also, the reliability of gossip rapidly increases with $B \cdot F$. In general, for a given value of $B \cdot F$, higher reliability is obtained by having a larger F (and therefore a smaller B). This is because a processor will have a more accurate idea of which processors already have m the later it forward m.

Table 2 shows the average number of messages sent per broadcast for the runs that were used to generate Table 1. The number of messages sent is bounded by $BF(n - f)$. For a given value of $F \cdot B$, fewer messages are sent for larger F. In this case, some processors learn that most other processors have already received the broadcast by the time they receive m $F - 1$ times, and so do not forward m to B processors.

Figure 1 illustrates how gracefully the measured reliability of gossip degrades as a function of f. This figure was generated using simulation with 10,000 broadcasts done for each value of f and having the f processors crash at the beginning of each run. The number of processors n is 32, $B = 4$, and $F = 3$. The measured reliability is 0.9999 for $f = 3$ crashed processors and is 0.869 for $f = 16$ crashed processors. Notice that the measured reliability is not strictly decreasing with f; when f is close to n, only a few processors need to receive the message for the broadcast to be successful. Indeed, when $f = n - 1$, gossip trivially has a reliability of 1.

4 Harary Graphs

In this section, we discuss the approach of imposing a Harary graph of certain connectivity on a network of processors and having each processor flood over that graph. The graph will be connected enough to ensure that the reliability of the flooding protocol will be acceptably large.

B	F	$B \cdot F$	$f = 0$	$f = 1$	$f = 2$	$f = 3$	$f = 4$
2	1	2	64.00	62.00	61.00	60.00	59.00
3	1	3	96.00	93.00	90.00	88.00	86.00
4	1	4	128.00	124.00	120.00	116.00	113.00
2	2	4	121.64	116.99	114.19	111.09	107.77
3	2	6	187.97	181.75	175.46	170.58	165.44
2	3	6	180.29	173.00	169.09	164.42	159.37
4	2	8	251.80	243.75	235.66	227.53	220.57
2	4	8	235.52	225.39	220.51	214.31	207.58
3	3	9	280.12	270.68	261.10	253.77	246.06
4	3	12	375.59	363.45	351.19	338.95	328.30
3	4	12	370.57	357.66	344.79	335.27	325.02
4	4	16	498.84	482.36	465.90	449.32	435.11

Table 2: Average number of messages per broadcast with 10^4 broadcasts. $n = 32$, $B_{initial} = \min(B, f + 1)$.

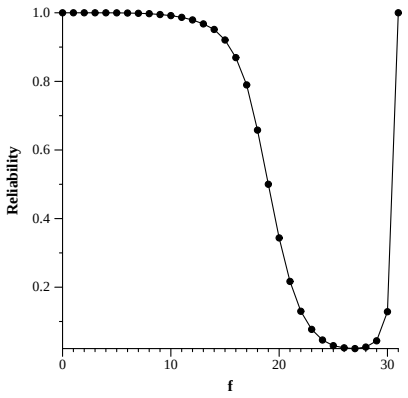

Figure 1: Reliability of gossip as a function of f $(B = 4, F = 3, n = 32)$.

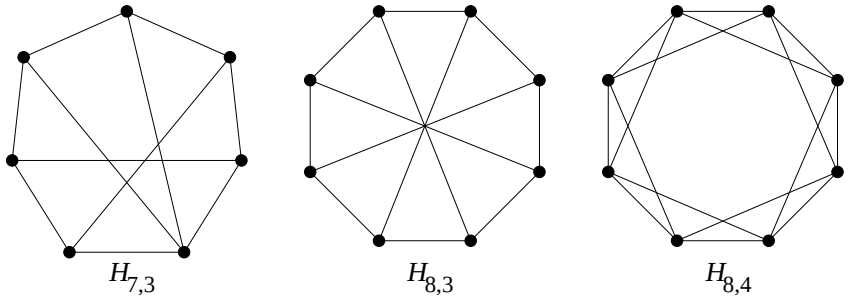

Figure 2: Graphs of $H_{7,3}, H_{8,3}$ and $H_{8,4}$.

4.1 Properties of Harary Graphs

A Harary graph is an $n-$node graph that satisfies the following three properties:

1. It is t-node connected. The removal of any subset of $t-1$ nodes will not disconnect the graph, but there are subsets of t nodes whose removal disconnects the graph.

2. It is t-link connected. The removal of any subset of $t-1$ links will not disconnect the graph, but there are subsets of t links whose removal disconnects the graph.

3. It is link minimal. The removal of any link will reduce the link connectivity (and therefore the node connectivity) of the graph.

Let $H_{n,t}$ denote the set of Harary graphs that contains n nodes and has a link and a node connectivity of t. For example, $H_{n,1}$ is the set of all $n-$node trees, and $H_{n,2}$ is the set of all $n-$node circuits. Figures 2 show examples of graphs in $H_{7,3}, H_{8,3}$ and $H_{8,4}$.

Harary gave an algorithm for the construction of a graph in $H_{n,t}$ for any value of n and $t < n$. [14] We denote this graph as $H^c_{n,t}$ and call it the *canonical Harary graph*. The algorithm is as follows:

$H^c_{n,1}$: The tree with edges $\forall i : 0 \leq i < n-1 : (i, i+1)$.

$H^c_{n,2}$: The circuit with edges $\forall i : 0 \leq i < n : (i, i+1 \bmod n)$.

$H^c_{n,t}$ for $t > 2$ and even: First construct $H^c_{n,2}$. Then, for each value of $m : 2 \leq m \leq t/2$ add edges (i, j) where $|i - j| = m \bmod n$.

$H^c_{n,t}$ for $t > 2$ and odd: First construct $H^c_{n,t-1}$. Then, connect all pairs (i, j) of nodes such that $j - i = \lfloor n/2 \rfloor$.

For most values of n and t, there are more than one graph in the set of $H_{n,t}$ graphs. The graphs given in Figure 2 are canonical Harary graphs, while the graph in Figure 3 is not (it is the unit cube).

It is not hard to see why Harary graphs are link-minimal among all t-connected graphs. For any graph G, the node connectivity $\kappa(G)$, link connectivity $\lambda(G)$, and minimum degree $\delta(G)$ are related:

$$\kappa(G) \leq \lambda(G) \leq \delta(G) \tag{1}$$

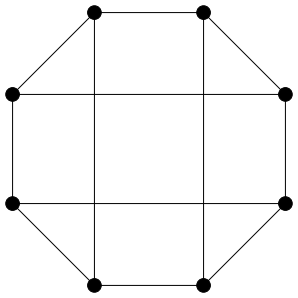

Figure 3: Another graph of $H_{8,3}$.

Harary showed that $\delta(G)$ is bounded by $\lceil 2\ell/n \rceil$, where ℓ is the number of links [14]. Therefore, to have t node or link connectivity, the number of links has to be at least $\lceil nt/2 \rceil$. If G is a regular graph (that is, all of the nodes have the same degree), then $\kappa(G) = \lambda(G) = \delta(G) = t$. A regular graph with $nt/2$ links is thus link-minimal among all graphs with t node or link connectivity. Harary graphs are such graphs when t is even or when n is even and t is odd.

When both n and t are odd, Harary graphs are not regular graphs because there is no regular graph of odd degree with an odd number of nodes. Rather, there are $n-1$ nodes of degree t and one node of degree $t+1$. The number of links is $\lceil nt/2 \rceil$. They are link-minimal because removing any link will result in at least one node having its node degree reduced from t to $t-1$.

4.2 Overhead and Reliability of Flooding on Harary Graphs

When n or t is even and assuming no failures, the overhead of flooding over a Harary graph is bounded from above by $n(t-1)+1$: the processor that starts the broadcast sends t messages and all the other processors send no more than $t-1$ messages. When n and t are odd one more message can be sent because the graph is not regular.

As long as the communications graph remains connected, a flooding protocol is guaranteed to be reliable. Hence, we characterize reliability as the probability of $H_{n,t}$ disconnecting. We first consider a failure model that includes the failure of nodes — that is, processor crashes, and the failure of links — that is, message channels that can drop messages due to congestion or physical faults. We then consider only node failures.

For local-area networks and small wide-area networks, one normally assumes that each link has the same probability p_ℓ of failing and each node has the same probability p_n of failing, and that all failures are independent of each other. We do so as well. This implies that dependent link failures like a processor holding carrier high or an internet router crashing are outside of our failure model.[1] Indeed, such failures cannot be masked without redundant networks and routers.

Since $H_{n,t}$ is both t-connected and t-link connected, one can compute an upper bound on the probability of $H_{n,t}$ being disconnected: the disconnection of $H_{n,t}$ requires x node failures and y link failures such that $x + y \geq t$. [3] The reliability r is thus bounded from below by the following:

$$r \geq 1 - \sum_{x=0}^{n-2} \binom{n}{x} p_n^x (1-p_n)^{n-x} \sum_{y=\max(t-x,0)}^{\ell} \binom{\ell}{y} p_\ell^y (1-p_\ell)^{\ell-y} \tag{2}$$

[1] Dependent link failures, like that of a network controller, that cause a processor to become isolated from the network can be characterized as node failures.

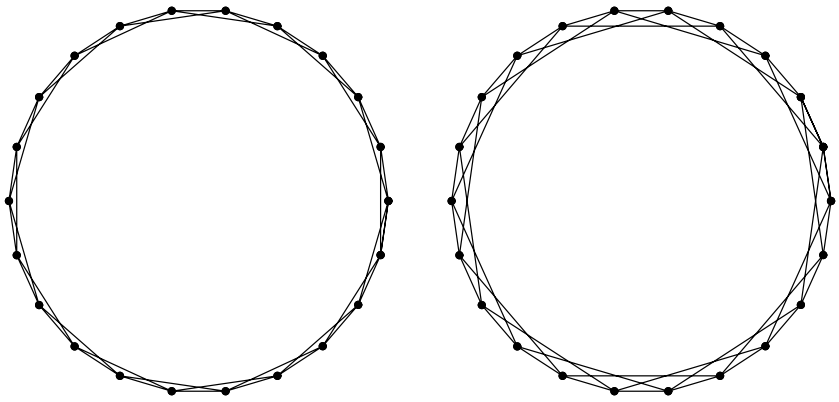

Figure 4: Two graphs of $H_{22,4}$.

where ℓ is the number of links in $H_{n,t} : \ell = \lceil nt/2 \rceil$. This formula, however, is conservative since it assumes that *all* such failures of nodes and links disconnect the graph. For example, consider $H_{4,2}$ and assume that $p_n = 0.99$ and $p_\ell = 0.999$ (that is, each processor is crashed approximately 14 minutes a day and a link is faulty approximately 1.4 minutes a day). The above formula computes a lower bound on the reliability of 0.9993. If we instead examine all of the failures that disconnect $H_{4,2}$ and sum the probabilities of each of these cases happening, then we obtain an actual reliability of 0.9997.

In general, computing the probability of a graph disconnecting given individual node and link failure probabilities is hard [6], and so using Equation 2 is the only practical method we know of for computing the reliability of flooding on $H_{n,t}$. But, if one assumes that links do not fail then one can compute a more accurate value of the reliability. Note that assuming no link failures may not be an unreasonable assumption. Most link failures in local-area networks and small wide-area networks are associated with congestion. Hence, they rarely endure and can often be masked by using a simple acknowledgment and retransmission scheme.

Consider the following metric:

Definition 1 *Given a graph $G \in H_{n,t}$, the* fragility $F(G)$ *of G is the fraction of subsets of t nodes whose removal disconnects G.*

For example, in the left graph of Figure 4, 187 of the 7,315 subsets of 4 nodes are cutsets of the graph, and so the graph has a fragility of 0.0256. The graph on the right, also a member of $H_{22,4}$, has a fragility of $22/7{,}315 = 0.0030$. The differences in fragility of the two graphs of Figure 4 is large. As we show below, the left-hand graph has a fragility of $n(n-t-1)/\left(2\binom{n}{t}\right)$, whereas the right-hand one has a fragility of $n/\binom{n}{t}$.

Since here we consider node failures only, p_ℓ is zero. We further assume that p_n is small. Thus, the probability of G disconnecting can be estimated as the probability of f nodes failing weighted by $F(G)$: $F(G)p_n^t(1-p_n)^{n-t}$. So,

$$r \leq 1 - F(G)p_n^t(1-p_n)^{n-t} \tag{3}$$

This is an upper bound on r because it does not consider disconnections arising from more than f nodes failing. But, when p_n is small the contribution due to more than f nodes failing is small.

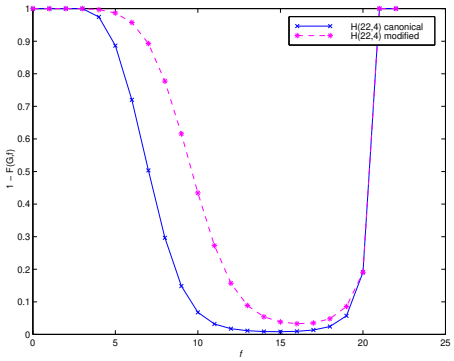

Figure 5: Graceful degradation of $H_{22,4}$. The dashed line is for the modified graph and the solid line for the canonical graph.

Assume that each processor is crashed for five minutes a day, and so $p_n = 0.0035$. From Equations 2 and 3 and setting $p_\ell = 0$ we compute $0.999998989 \leq r \leq 0.999999974$ for flooding in the left graph of Figure 4. In fact, the reliability r computed by enumerating all the cutsets and summing their probability of occurring is 0.999999973. Similarly, for the right-hand graph of Figure 4, we compute $0.999998989 \leq r \leq 0.999999997$ and r is 0.999999997.

4.3 Graceful Degradation

Still assuming that $p_\ell = 0$, one can extend the notion of fragility to compute how gracefully reliability degrades in flooding on a Harary graph.

Definition 2 *Given a graph $G \in H_{n,t}$, $F(G, f)$ is the fraction of subsets of f nodes whose removal disconnects G.*

Hence, $F(G, t) = F(G)$ and $F(G, f) = 0$ for $0 \leq f < t$ or $n - 1 \leq f \leq n$. On the condition of any subset of f nodes having failed, the graph will remain connected with the probability of $1 - F(G, f)$. Thus, we can use $1 - F(G, f)$ as a way to characterize the reliability of flooding on a Harary graph G. This is much like the way we measured the reliability of gossip as a function of f as illustrated in Figure 1. There, the reliability was measured after f nodes have crashed.

The graphs of $1 - F(G, f)$ for the two graphs of Figure 4 is shown in Figure 5. As can be seen, the less fragile graph also degrades more gracefully than the canonical graph.

4.4 Bounds on Fragility

Since the upper bound on reliability and how gracefully reliability degrades depend on the fragility of a Harary graph, we examine some fragility properties of canonical Harary graphs. We show that canonical Harary graphs can be significantly more fragile than some other families of Harary graphs, and describe how to construct these less fragile graphs.

In the following, the symbol $\oplus$ signifies addition modulo n and $\ominus$ subtraction modulo n. Given a node i, the set of nodes $\{i, i \oplus 1, i \oplus 2, \ldots, i \oplus k\}$ for some $k \geq 0$ are *sequential* to each other, and the node $i + n/2$ the *antipodal* node of i. A single node can be thought of as a one-element set of sequential nodes.

We define a $t-$cutset of a graph G to be a set of t nodes of G whose removal disconnects G. And, given a subgraph S of G, we define the joint neighbors of S to be those nodes in $G-S$ that are a neighbor of a node in S. The two ideas are related: if S is connected, A are the joint neighbors of S, and $A \cup S \subset G$, then A is a $|A|-$cutset of G.

$H_{n,n-1}$ is an $n-$clique and so $F(H_{n,n-1}, f) = 0$ for all f between 0 and n. Hence, in the following we assume that $t < n-1$.

Lemma 1 *For $H^c_{n,t}$ with even t, the set of joint neighbors of any sequence of nodes $A = \{i, i \oplus 1, \ldots, i \oplus z\}$ has a size of t as long as $z < n - t$.*

Proof: Since $H^c_{n,t}$ for even n or t is symmetric, we can assume without loss of generality that $i = 0$. From the definition of $H^c_{n,t}$, the joint neighbors of A are $\{n - t/2, n - t/2 + 1, \ldots, n-1, z+1, z+2, \ldots, z + t/2\}$. These nodes are distinct as long as $z - t/2 < n - t/2$, or as long as $z < n - t$. If this holds, then the number of joint neighbors is t.
□

Lemma 2 *For $H^c_{n,t}$ with even $t > 2$, consider any two nodes i and $i \oplus (k+1)$ for $k > 0$. If $k < t$, then all of the nodes between i and $i \oplus k$ are joint neighbors of i and $i \oplus (k+1)$; otherwise, t of these nodes are joint neighbors of i and $i \oplus (k+1)$.*

Proof: Without loss of generality, let $i = 0$. From the definition of $H^c_{n,t}$, the nodes $\{1, 2, \ldots, t/2\}$ are all neighbors of 0 and the nodes $\{k - t/2 + 1, k - t/2 + 2, \ldots, k\}$ are all neighbors of $k+1$. If $k - t/2 + 1 \leq t/2$ (which is equivalent to $k < t$) then these two sets intersect and their union contains k nodes. Otherwise, the union of these two sets contains t nodes.
□

Lemma 3 *For any subset A of $H^c_{n,t}$ such that A does not consist of sequential nodes and $1 < |A| \leq n - t - 2$, the set of joint neighbors of A contains at least $t+1$ nodes.*

Proof: We only consider subgraphs A of $H^c_{n,t}$ that have no more than $\lfloor (n-t)/2 \rfloor$ nodes. Any larger subgraph A' with a $t-$cutset shares that cutset with a subgraph A whose size is $n - t - |A'|$ which is less than $\lfloor (n-t)/2 \rfloor$.

A consists of $g > 1$ sets of sequential nodes. Let $H_1, H_2, \ldots, H_g$ be the corresponding sets of sequential nodes in $H^c_{n,t} - A$ (which we call the "holes"). Without loss of generality, let H_1 be hole with the largest size. Since $g > 1$ there is a second hole H_2. We consider two cases:

1. $|H_1| \geq t$. By Lemma 2, H_1 contributes t joint neighbors to A, and by the same lemma H_2 contributes at least one joint neighbor to A. Hence, there are at least $t+1$ joint neighbors of A.

2. $|H_1| < t$. By Lemma 2, each of the remaining $g-1$ holes contributes at least 1 joint neighbor to A. Since H_1 is one of the largest holes, $\forall i : 1 \leq i \leq g : |H_i| < t$. By Lemma 2, each other hole H_i contributes $|H_i|$ to the joint neighbors of A. Hence, there are $\sum_{i=1}^{g} |H_i|$ joint neighbors of A.

   By definition, $n = |A| + \sum_{i=1}^{g} |H_i|$, and so $\sum_{i=1}^{g} |H_i| = n - |A| \geq n - \lfloor (n-t)/2 \rfloor$. This implies $\sum_{i=1}^{g} |H_i| \geq \lceil (n+t)/2 \rceil$. Since $n > t$, $\lceil (n+t)/2 \rceil > t$ and so A has more than t joint neighbors.

In both cases, the lemma holds.
□

Lemma 3 considers cutsets for

Theorem 1 *For even $t > 2$ and $n \geq t+2$, the number of subsets of t nodes that disconnects $H^c_{n,t}$ is $n(n-t-1)/2$.*

Proof: Lemmas 1 and 3 together show that for $H^c_{n,t}$ with even t, the only sets of nodes that have a set of joint neighbors of size t are sequential nodes of size no larger than $n-t$. They also imply that each $t-$cutset divides the graph into two sequential sets of nodes.

Let $A = \{i, i \oplus 1, \ldots, i \oplus z\}$ be a set of sequential nodes, C its joint neighbors, and $B = G - C - A$ the remaining set of $n-t-z-1$ sequential nodes.

For any value of z, there are n pairs of A and B, one pair for each value of $i : 0 \leq i \leq n-1$. And, for any value of z, as long as $|A| < |B|$, we can count n cutsets without double counting. When $|A| = |B|$, we can count $n/2$ cutsets without double counting.

Thus we count n cutsets for all integer values of $z : z \geq 0$ and $z+1 < n-t-z-1$. Simplifying, we get $0 \leq z < (n-t-2)/2$. Similarly, we count $n/2$ cutsets when $z = (n-t-2)/2$ is an integer.

If n is odd, then $(n-t-2)/2$ is not an integer and the largest integer value of z satisfying $z < (n-t-2)/2$ is $(n-t-3)/2$. Hence, we count $n((n-t-3)/2+1) = n(n-t-1)/2$ cutsets. If n is even, then $(n-t-2)/2$ is an integer and the largest integer value of z satisfying $z < (n-t-2)/2$ is $(n-t-2)/2-1$. Hence, we count $n((n-t-2)/2-1+1+1/2) = n(n-t-1)/2$ cutsets.
□

We can construct a member of $H_{n,t}$ for even t and $n > 2t$ that are less fragile than $H^c_{n,t}$. These graphs, which we call *modified* Harary graphs and denote with $H^m_{n,t}$, have n distinct $t-$cutsets. The right hand graph of Figure 4 is $H^m_{22,4}$, while the left hand graph is $H^c_{22,4}$. An algorithm for constructing $H^m_{n,t}$ is as follows:

$H^m_{n,t}$: First construct $H^c_{n,2}$. Then, for each value of m: $2 \leq m \leq t/2$ add edges (i,j) where $|i-j| = (m+1) \bmod n$.

Lemma 4 *For $H^m_{n,t}$, the set of joint neighbors of any sequence of nodes $A = \{i, i \oplus 1, \ldots, i \oplus z\}$ has size t when $z = 0$ or $z = n-t-1$, size $t+1$ when $z = n-t-2$, and size $t+2$ when $0 < z < n-t-2$.*

Proof: Since $H^m_{n,t}$ is symmetric, we can assume without loss of generality that $i = 0$. From the definition of $H^m_{n,t}$:

- When $z = 0$, the neighbors of 0 are $\{1, 3, 4, \ldots, t/2+1, n-1, n-3, n-4, \ldots, n-t/2-1\}$. These nodes are unique as long as $n - t/2 - 1 > t/2 + 1$ which can be rewritten as $n > t+2$. Since $H^m_{n,t}$ is defined only for $n > 2t$ and for even $t > 2$, $n > t+2$ holds.
- When $z > 0$ then the joint neighbors of A are $\{z+1, z+2, z+3, \ldots, z+t/2+1, n-1, n-2, n-3, n-4, \ldots, n-t/2-1\}$. If $z + t/2 + 1 < n - t/2 - 1$ then this set contains $t+2$ nodes. This inequality can be rewritten as $z < n-t-2$. If $z + t/2 + 1 = n - t/2 - 1$ (which can be written as $z = n-t-2$) then this set contains $t+1$ nodes. Similarly, if $z = n-t-1$ then this set contains t nodes.

These two cases establish the lemma.
□

Lemma 5 *For $H^m_{n,t}$ consider any two nodes i and $i \oplus (k+1)$ for $k > 0$. $C(k)$ of the nodes between i and $i \oplus (k+1)$ are joint neighbors of $\{i, i \oplus (k+1)\}$ where*

$$C(k) = \begin{cases} t & k > t+1 \\ k-2 & t/2+3 \leq k \leq t+1 \\ k & 4 \leq k \leq t/2+2 \\ 2 & 2 \leq k \leq 3 \\ 1 & k = 1 \end{cases}$$

Proof: Without loss of generality we can set $i = 0$; hence, $C(k)$ is not a function of i. From the definition of $H^m_{n,t}$, $\{1, 3, 4, \ldots, t/2+1\}$ are neighbors of 0 and $\{k - t/2, \ldots, k-3, k-2, k\}$ are neighbors of $k+1$. If $t/2+1 < k - t/2$ then all of these t nodes are distinct. For smaller values of k through $k - t/2 = 3$, only $k-2$ of these nodes are distinct and in the range of 1 through k. For yet smaller values of k through $k = 4$ all k of the nodes in the range of 1 through k are distinct. The other two cases follow directly from substituting the corresponding value of k.
□

Lemma 6 *For any subset A of $H^m_{n,t}$ such that A is not a single set of sequential nodes and $1 < |A| \leq \lfloor (n-t)/2 \rfloor$, the set of joint neighbors of A contains at least $t+1$ nodes.*

Proof: A consists of $g > 1$ sets of sequential nodes. Let $H_1, H_2, \ldots, H_g$ be the corresponding sets of sequential nodes in $H^m_{n,t} - A$ (the "holes"). Thus, $\lceil (n+t)/2 \rceil \leq \sum_{i=0}^{g} |H_i| \leq n-2$. Without loss of generality, let H_1 be hole with the largest size. There is at least one more hole H_2. There are two cases:

1. $|H_1| > t+1$. By Lemma 5, H_1 contributes t joint neighbors to A, and by the same lemma H_2 contributes at least one joint neighbor to A. Hence, there are at least $t+1$ joint neighbors of A.

2. $|H_1| \leq t+1$. The number of joint neighbors of A is $\sum_{i=1}^{g} C(|H_i|)$. From Lemma 5, when $t \geq 6$ $C(|H_i|)/|H_i| \geq 2/3$ (the lower bound is met when $|H_i| = 3$). There are two cases:

   (a) $t = 4$: We consider the possible values of $|H_1|$ and use the fact that $\lceil (n+4)/2 \rceil \leq \sum_{i=1}^{g} |H_i| \leq n-2$. If $|H_1| = 5$ then H_1 contributes 3 joint neighbors. From the definition of $H^m_{n,t}$, $n \geq 9$ and so $7 \leq |H_1| + \sum_{i=2}^{g} |H_i|$ or $\sum_{i=2}^{g} |H_i| \geq 2$. Hence, the remaining holes contribute 2 joint neighbors bringing the total to 5. If $|H_1| = 4$ then H_1 contributes 4 joint neighbors, $\sum_{i=2}^{g} |H_i| \geq 3$ and the remaining holes contribute at least 2 joint neighbors. If $|H_1| = 3$ then H_1 contributes 2 joint neighbors, $\sum_{i=2}^{g} |H_i| \geq 4$ and the remaining holes contribute at least 3 joint neighbors. If $|H_1| = 2$ then H_1 contributes 2 joint neighbors, $\sum_{i=2}^{g} |H_i| \geq 5$ and the remaining holes contribute at least 3 joint neighbors. Finally, if $|H_1| = 1$ then H_1 contributes 1 joint neighbor, $\sum_{i=2}^{g} |H_i| \geq 6$ and the remaining holes contribute at least 4 joint neighbors. In all cases there are at least 5 joint neighbors of A.

(b) $t > 4$: $\sum_{i=1}^{g} C(|H_i|) \geq 2/3 \sum_{i=1}^{g} |H_i|$. From above, $\sum_{i=1}^{g} |H_i| \geq \lceil (n+t)/2 \rceil$ which is at least $(n+t)/2$. Hence, there are at least $(n+t)/3$ joint neighbors of A. By definition of $H^m_{n,t}$, $n > 2t$ hence $(n+t)/3 > t$. Thus, there are more than t joint neighbors of A.

In both cases, the lemma holds.
□

Theorem 2 *When $n > 2t$, $H^m_{n,t}$ has n $t-$cutsets.*

Proof: From Lemma 4, $H^m_{n,t}$ has at least n $t-$cutsets, namely those for each value of i and when $z = 0$. The other value of z which results in a set of joint neighbors of size t is when A has $n - t$ nodes, and so the removal of these nodes will not disconnect $H^m_{n,t}$. Lemma 6 shows that any other subset of $H^m_{n,t}$ has at least a $t+1-$cutset. Hence there are only n $t-$cutsets.
□

Theorem 2 was shown by showing that all subsets of nodes of $H^m_{n,t}$ have at least t joint neighbors. Hence, $H^m_{n,t}$ is $t-$node connected. It is easy to see that each node has degree t and that the graph has $\lceil nt/2 \rceil$ edges. Hence, by Equation 1 $H^m_{n,t}$ is in $H_{n,t}$.

It is not hard to find examples of Harary graphs that have fewer than n $t-$cutsets. For example, consider a bipartite graph with t nodes on each side and with each node on one side connected to each node on the other side. It is easy to see that this graph has two $t-$cutsets, no cutsets smaller than t, a degree of t, and t^2 edges. Hence, it is in $H_{2t,t}$. Or, consider a Harary graph $G \in H_{n,t}$ that has c $t-$cutsets. Construct the graph G' as follows:

1. For every node in G, create k nodes in G'.
2. For every edge (u, v) in G, add the k^2 edges connecting each of the nodes associated with u with each of the nodes associated with v.

The graph G' has degree kt and $\lceil k^2 nt/2 \rceil$ edges. It can be disconnected only by deleting the kt nodes associated with a $t-$cutset of G. Hence, G' is in $H_{kn,kt}$ and has c $kt-$cutsets. Thus, for example, instead of using $H^m_{100,16}$ which has 100 $16-$cutsets, one can construct a graph by replicating each node in $H^m_{25,4}$ four times. This results in a graph that has 25 $16-$cutsets, which is four times less fragile than $H^m_{100,16}$. For arbitrary n and t, however, it remains an open problem of what is the minimum number of cutsets of size t for graphs in $H_{n,t}$.

Canonical Harary graphs have a better fragility when t is odd, which we now show. We first show that when $t = 3$, $H^c_{n,t}$ has n $3-$cutsets for even $n > 6$ or odd $n > 7$. Then, we show that for $t \geq 5$ and $n > t + 2$, there are n $t-$cutsets in $H^c_{n,t}$.

Theorem 3 *For $H^c_{n,t}$ with even $n > 6$ and $t = 3$, there are n cutsets of size t.*

Proof: Since every node in $H^c_{n,3}$ has degree 3, each node has a $3-$cutset. For two nodes i, j to have the same set of neighbors, there is (with no loss of generality) some node k such that $i+1 = k = j-1$ and another node m such that $i - 1 = m = j + n/2$. Thus, $j = i + 2$ and $j + n/2 = i - 1 = n + i - 1$. Rewriting the first equation gives $j - i = 2$ and rewriting the second equation gives $j - i = n/2 - 1$. Combining, we get $2 = n/2 - 1$ or $n = 6$. Thus, if $n > 6$ then all nodes have distinct sets of neighbors. Hence, when $n > 6$ and even, there are at least n cutsets of size 3.

When $n > 6$ and even, a set of connected nodes A is either a set of sequential nodes or a set of sequential nodes and a subset of their corresponding antipodal nodes. If A is a set of sequential nodes $\{i, \ldots, i \oplus k\}$ where $1 \leq k \leq \lfloor (n-t)/2 \rfloor$, then the number of joint neighbors is at least four: $\{i \ominus 1, i \oplus (k+1)\}$ and the k corresponding antipodal nodes. If A is a set of sequential nodes plus a subset of their antipodal nodes, then again the number of joint neighbors is at least four. If only one antipodal node is in A, then the two neighbors of this node are joint neighbors. If there are at least two antipodal nodes, then at least two of them will contribute a neighbors to the joint neighbor set.
□

Theorem 4 *For $H^c_{n,t}$ with odd $n > 7$ and $t = 3$, there are n cutsets of size t.*

Proof: In $H^c_{n,t}$, node $(n-1)/2$ has four neighbors; the rest have three neighbors. Each node except $(n-1)/2$ has a cutset of size 3, and the subset $\{0, n-1\}$ has the cutset $\{1, (n-1)/2, n-2\}$ which has a size 3.

An argument similar to the previous proof says all nodes except $(n-1)/2$ have distinct neighbors. The only nodes that can have same cutset as $\{0, n-1\}$ are 2 and $n-3$. If 2 has the same cutset, that is, $\{1, (n-1)/2, n-2\} = \{1, 3, 2+(n-1)/2\}$, then we have $(n-1)/2 = 3$ or $n = 7$. A similar argument holds for $n-3$. Hence, if $n > 7$ then $H^c_{n,3}$ has n distinct cutsets of size 3.
□

Theorem 5 *For $H^c_{n,t}$ with even $n \geq t+2$ and odd $t \geq 5$, there are n cutsets of size t. For $H^c_{n,t}$ with odd $n > t+2$ and odd $t \geq 5$, there are n cutsets of size t.*

Proof: $H^c_{n,t}$ for odd t has embedded within it $H^c_{n,t-1}$. From the definition of $H^c_{n,t}$, each node (except $\lfloor n/2 \rfloor$ when n is odd) has t neighbors that are distinct. When n is odd, nodes $\{0, n-1\}$ have as joint neighbors the t nodes $\{1, 2, \ldots, (t-1)/2, n-2, n-3, \ldots, n-(t-1)/2-1, (n-1)/2\}$. Hence, when n is even and $n \geq t+2$ then there are at least n $t-$cutsets. If n is odd, $\{0, n-1\}$ is not separable by removing its t joint neighbors unless $n > t+2$, in which case there again are at least n $t-$cutsets.

We now show that there are no other $t-$cutsets. We only consider the cutsets for subgraphs A of $H^c_{n,t}$ that have no more than $\lfloor (n-t)/2 \rfloor$ nodes. Any larger subgraph A' with a $t-$cutset shares that cutset with a graph A whose size is within our range of consideration.

First consider a subgraph of $H^c_{n,t}$ that consists of $z+1$ sequential nodes for $z > 1$. Let A be the nodes of this subgraph. By Lemma 1 A has $t-1$ joint neighbors not considering the antipodal nodes of A. Let i be the first node of A. Since $\lfloor n/2 \rfloor > \lfloor (n-t)/2 \rfloor$ the antipodal nodes of the first and last nodes of A are not in A; indeed, none of of the antipodal nodes of A are in A. And, the antipodal nodes of all nodes (except for 0 and $n-1$ when n is odd) are distinct. Hence, as long as n is even or A is not $\{0, n-1\}$, it has at least $t+1$ joint neighbors.

Now consider a subgraph of $H^c_{n,t}$ whose nodes A are not a single sequential set of nodes. Hence, A consists of $g > 1$ sets of sequential nodes. As in the previous proofs, let H_i represent the corresponding $g > 1$ "holes", which are the sets of sequential nodes in the nodes of $H^c_{n,t}$ minus A. As before, we assume without loss of generality that H_1 has the largest size. There are three cases:

1. $|H_1| > t-1$. Let $H_1 = \{i \oplus 1, i \oplus 2, \ldots, i \oplus k\}$ for some $k > t-1$. By the definition of $H^c_{n,t}$, the nodes $\{i \oplus 1, i \oplus 2, \ldots, i \oplus (t-1)/2\}$ are neighbors of i and $\{i \oplus (k-(t-1)/2+1), i \oplus (k-(t-1)/2+$

$2), \ldots, i \oplus k\}$ are neighbors of $i \oplus (k+1)$. Thus, we have counted $t-1$ joint neighbors of A, and the remaining $k-t+1$ nodes of H_1 are $\{i \oplus ((t-1)/2+1), i \oplus ((t-1)/2+2), \ldots, i \oplus (k-(t-1)/2)\}$. Denote this set as M.

Since $g > 1$, there is at least one other hole. If there is more than one node in the remaining holes, then by Lemma 2 we could add at least two more joint neighbors to A bringing the total up to at least $t+1$. Hence, for contradiction, we assume that there is one more hole $H_2 = \{h\}$. This hole contributes one more joint neigbor, bringing the total up to t. And, since $\sum_{i=1}^{g} |H_i| \geq \lceil (n+t)/2 \rceil$, $|H_1| \geq \lceil (n+t)/2 \rceil - 1$. In particular, $|H_1| > n/2$, and so the antipodal nodes of A are in H_1; in particular, they overlap the middle of H_1, where M is.

Every node of A that has an antipodal node in M contributes one more joint neighbor to A. M must contain at least one node, since otherwise $|H_1| \leq t-1$. If A is to have only t joint neighbors, then M can contain only one node and whose antipodal node is not in A. Thus $|H_1| = t$.

We thus have the following scenario: $|A| = n-t-1$ (that is, it contains all of the nodes not in H_1 and H_2) and it may have a $t-$cutset. If it does, then removing these t nodes results in the two disconnected graphs A and a graph containing a single node. Hence, if there is such a cutset, then it was counted above as one of the n $t-$cutsets of $H^c_{n,t}$.

2. $|H_1| = t-1$. Since $\sum_{i=2}^{g} |H_i| \geq \lceil (n+t)/2 \rceil - |H_1|$, we have $\sum_{i=2}^{g} |H_i| \geq (n-t+1)/2$. This latter quantity is greater than 1 when $n > t+1$, that is when $H^c_{n,t}$ is not a clique. By Lemma 2, H_1 contribute $t-1$ joint neighbors of A, and from Lemma 2 and $\sum_{i=2}^{g} |H_i| \geq 2$ the remaining holes contribute at least two joint neighbors of A. Hence, A has more than t joint neighbors.

3. $|H_1| < t-1$. From Lemma 2, each H_i contributes $|H_i|$ neighbors of A. Since $\sum_{i=1}^{g} |H_i| \geq \lceil (n+t)/2 \rceil$ and $\lceil (n+t)/2 \rceil > t$, A has more than t joint neighbors.

Thus, the only $t-$cutsets are the n enumerated above.
□

When n and t are both odd, only $n-1$ nodes have t neighbors; one node has $t+1$ neighbors. Hence, it may be surprising that $H^c_{n,t}$ for odd n and t has n $t-$cutsets. It is not hard to construct graphs in $H_{n,t}$ for odd n and t that have $n-1$ $t-$cutsets. For example, Figure 6 gives a graph in $H_{9,3}$ that has 8 $3-$cutsets. The improvement in fragility for such graphs, however, is small.

For a fixed $t > 2$, the fragility $F(H^c_{n,t})$ decreases with increasing n. This is because the number of cutsets of size t grows either linearly or quadratically with n, while $\binom{n}{t}$ is of $O(n^t)$. Similarly, $F(H^m_{n,t})$ decreases with increasing n. The same observations hold for $F(H^c_{n,t})$ and $F(H^c_{n,t})$ when n is fixed and t is increased.

5 Comparing Harary Graph-Based Flooding and Gossip

A simple comparison of flooding over a Harary graph and gossip indicates that each has its own strengths:

- *scalability*

  The two protocols are equivalent. Both protocols run simple algorithms that are based on slowly-changing information. Both require a processor to know the identity of the other

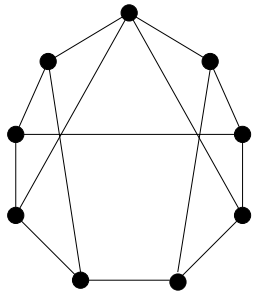

Figure 6: A less fragile $H_{9,3}$.

processors on its (local-area or small wide-area) network and a small amount of constant information (the rule used to identify neighbors versus the constants B and F). And, in both protocols the number of messages that each processor sends is independent of n.

- *adaptability*

  The flooding protocol requires the processors to use the same Harary graph. Since each processor independently determines its neighbors, it might appear that gossip is more adaptable than flooding over a Harary graph. From a practical point of view, though, we expect that they are similar in adaptability. Given the controlled environment of local-area networks and small wide-area networks, it is not hard to bound from below with equivalent reliabilities the time it takes for each protocol to terminate. Then, one can use reliable broadcast based on the old set of processors to disseminate the new set of processors. In addition, adding or removing a single processor causes only t processors to change their neighbors in Harary graph flooding. If t is small, then a non-scalable agreement protocol can be used to change the set of processors.

- *graceful degradation*

  Gossip degrades more gracefully than Harary graph flooding. Figure 7 illustrates this for $n = 22$. The Harary graphs used here are the $H^c_{22,4}$ and $H^m_{22,4}$, and the gossip protocols use $B = 3, F = 2$ and $B = 2, F = 2$, both with $B_{initial} = B$. We compare the degradation of reliability of these protocols because they have similar message overheads. Note that while the Harary graph flooding yields a *higher* reliability for small f even when $f > t$, both gossip protocols have a higher reliability for $f > 10$.

- *message overhead*

  Since $H_{n,t}$ has a minimum number of links while remaining t-connected, it is not surprising that Harary graph flooding sends fewer messages than gossip. For example, given $n = 32$, flooding on $H_{32,4}$ and gossip with $B = 4$ and $F = 3$ provide similar reliabilities given processors that are crashed for five minutes a day. Gossip sends roughly $BF/(t-1) = 4$ times as many messages as Harary graph flooding.

To make a more detailed comparison of message overhead, we evaluated the performance of gossip and Harary graph flooding using the *ns* simulator [25]. We simulated Ethernet-based networks. One of the networks we considered was a LAN of a single Ethernet, where there are 32 processors.

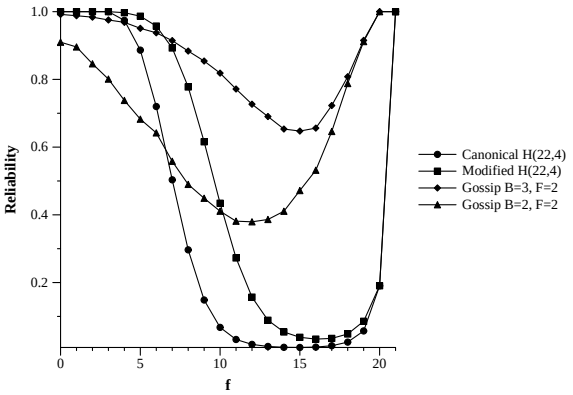

Figure 7: Graceful degradation of gossip and flooding on $H_{22,4}$

The other was a small WAN of three Ethernets pairwise connected, where each Ethernet has 21 processors, one of them also acting like a router.

For the single LAN we imposed the $H^m_{32,4}$ graph on the processors, and for the small WAN $H^m_{63,4}$ was imposed where processors 0, 21 and 42 were the routers connecting the three LANs. Thus on each LAN, the router has two neighbors on a different LAN and another processor has one neighbor outside of the LAN. Flooding on these Harary graphs was compared with gossip with $B = 4, F = 3$, and $B_{initial} = B$.

We obtained the properties of an Ethernet based on those of a common Ethernet for LANs. For the single LAN, we assume a bandwidth of 10 Mbps, and for the small WAN, we assumed that the links between routers of the LANs have a bandwidth of 1 Mbps and a delay of 10 ms. The ns simulator followed the Ethernet specifications and provided the low-level details. Each message was contained in a 1K-byte packet. We did not consider failures in these simulations. All the results shown are the average of 100 broadcasts.

Figure 8 shows that gossip initially delivers faster. However, it takes longer for the last few processors to receive the message. Therefore, it takes longer for gossip to complete a broadcast than for Harary graph flooding. For both protocols, the completion time would be shorter if there were no collisions. There were a fair amount of collisions because both protocols make intensive use of the network.

In Figure 9, we plot the number of packets on the Ethernet histogrammed into 5-ms time slot. All the packets delivered after the completion time shown in Figure 8 are redundant packets. As can be seen, the packet flow of Harary graph flooding diminishes much more quickly than that of gossip. This shows that, as discussed above, gossip has a much higher message overhead.

In the small WAN, Harary graph flooding imposes a much smaller load on the routers than gossip does. We measured the number of packets that were sent across the LANs. With gossip, an average of 169 packets were sent between each pair of LANs. With Harary graph flooding, only 6 packets went across because processors on each LAN have a total of three neighbors on another LAN. This kind of overloading of routers with redundant messages happens when the network topology is not taken into account [20, 18].

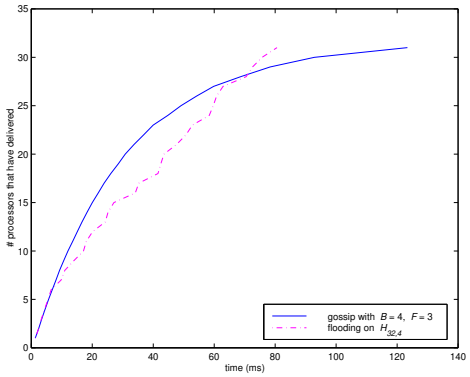

Figure 8: Completion time of gossip with $B = 4, F = 3$ and flooding on $H_{32,4}$

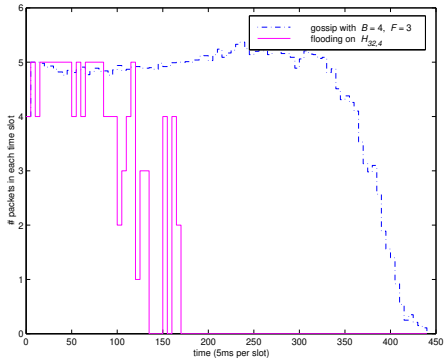

Figure 9: Number of packets per time slot of gossip with $B = 4, F = 3$ and flooding on $H_{32,4}$

6 Conclusions

Gossip protocols are often advertised as being attractive because their simplicity and their use of randomization makes them scalable and adaptable. Furthermore, their reliability degrades gracefully with respect to f. We believe that, while these advertised attractions are valid, Harary graph flooding also provides most of these attractions with a substantially lower message overhead. Furthermore, Harary graph flooding appears to be faster than gossip for broadcast-bus networks. Hence, for local-area networks and small wide-area networks, the only benefit of gossip is that its reliability more gracefully degrades than Harary graph flooding.

This remaining advantage of gossip, however, should be considered carefully. While it is true that gossip more gracefully degrades than Harary graph flooding, the reliability of flooding over $H^m_{n,t}$ decreases only slightly for values of f slightly larger than t. Hence, Harary graph flooding provides some latitude in computing t. And, both suffer from reduced reliability as f increases further. One improves graceful degradation of gossip by increasing F (or, to a slightly less degree, B). Doing so increases the message overhead and network congestion, and therefore the protocol completion time.

References

[1] G. R. Andrews. *Concurrent programming: principles and practice*, Benjamin/Cummings, 1991.

[2] Ö. Babaoğlu. On the reliability of consensus-based fault-tolerant distributed computing systems. *ACM Transactions on Computer Systems*, 5:394–416, 1987.

[3] L. W. Beineke and F. Harary. The connectivity function of a graph. *Mathematika* 14(1967), pp. 197–202.

[4] K. Birman, *et al.* Bimodal multicast. Cornell University, Department of Computer Science Technical Report TR-98-1665, May 1998.

[5] M. Clegg and K. Marzullo. A Low-cost processor group membership protocol for a hard real-time distributed system. In *Proceedings of the 18th IEEE Real-Time Systems Symposium*, 1997, pp. 90–98.

[6] C. J. Colbourn. *The combinatorics of network reliability*, Oxford University Press, 1987.

[7] F. Cristian. Synchronous atomic broadcast for redundant broadcast channels. *Real-Time Systems* 2(1990), pp. 195–212.

[8] F. Cristian and C. Fetzer. The timed asynchronous distributed system model. IEEE Transactions on Parallel and Distributed Systems, June 1999, 10(6):642–657.

[9] A. Demers, *et al.* Epidemic algorithms for replicated database maintenance. In *Proceedings of 6th ACM Symposium on Principles of Distributed Computing*, Vancouver, British Columbia, Canada, 10–12 August 1987, pp. 1–12.

[10] S. Floyd, *et al.* A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking* 5(6):784–803, December 1997.

[11] R. Friedman and S. Manor and K. Guo. Scalable stability detection using logical hypercube. Technion, Department of Computer Science Technical Report 0960, May 1999.

[12] R. A. Golding and D. E. Long. The performance of weak-consistency replication protocols. University of California at Santa Cruz, Computer Research Laboratory Technical Report UCSC-CRL-92-30, July 1992.

[13] K. Guo, *et al.* GSGC: an efficient gossip-style garbage collection scheme for scalable reliable multicast. Cornell University, Department of Computer Science Technical Report TR-97-1656, December 3 1997.

[14] F. Harary. The maximum connectivity of a graph. In *Proceedings of the National Academy of Sciences*, 48:1142–1146, 1962.

[15] M. G. Hayden and K. P. Birman. Probabilistic broadcast. Cornell University, Department of Computer Science Technical Report TR-96-1606, September 1996.

[16] T. Abdelzaher and A. Shaikh and F. Jahanian and K. Shin. RTCAST: Lightweight multicast for real-time process groups. In *Proceedings of the Second IEEE Real-Time Technology and Applications Symposium*, 1996, pp.250–259.

[17] A. Liestman. Fault-tolerant broadcast graphs. *Networks* 15:159–171, 1985.

[18] M. -J. Lin and K. Marzullo. Directional gossip: gossip in a wide area network. To be published in *Proceedings of the Third European Dependable Computing Conference (Springer-Verlag LNCS)*.

[19] B. Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, 47(1987):213–223.

[20] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. In *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98)*, The Lake District, England, September 1998, pp. 55-70.

[21] Fred B. Schneider. Byzantine generals in action: implementing fail-stop processors. *ACM Transactions on Computer Systems* 2(1984), pp. 145–154.

[22] Amitabh Shah. Exploring Trade-offs in the Design of Fault-Tolerant Distributed Databases. Ph.D. dissertation, Cornell University Department of Computer Science, August 1990.

[23] D. B. Terry, *et al.* Managing update conflicts in Bayou, a weakly connected replicated storage system. In *Proceedings of the 15th Symposium on Operating System Principles*, 1995, pp. 3–6.

[24] V. Hadzilacos and S. Toueg. Fault-tolerant broadcasts and related problems. In *Distributed Systems* (S. Mullender, editor), ACM Press, 1993.

[25] Network Simulator. http://www-mash.cs.berkeley.edu/ns.