Optimal broadcasting in networks with faulty nodes

Yoel Grinshpon       Ori Gurel-Gurevich

September 15, 2017

Abstract

Large computer networks are an essential part of modern technology, and quite often information needs to be broadcast to all the computers in the network. If all computers work perfectly all the time, this is simple. Suppose, however, that some of the computers fail occasionally. What is the fastest way to ensure that with high probability all working computers get the information?

In this paper, we analyze three algorithms to do so. All algorithms terminate in logarithmic time, assuming computers fail with probability $1 - p$ independently of each other. We prove that the third algorithm, which runs in time $(1 + o(1))\left(\frac{\log N}{\log(1+p)}\right)$, is asymptotically optimal.

Keywords. Rumor spreading, Randomized broadcasting, fault tolerant broadcasting.

1 Introduction and results

Suppose we have a network with $N$ nodes (which stand for computers), each of them active independently with probability $p$. Suppose also that one of these nodes has a message that needs to be conveyed to all active nodes. Each node can send one message per time unit to any other node. Once an active node receives the message, it too can send the message to other nodes. The question we are asking is what method of spreading the message will minimize the time we have to wait until all active nodes have received the
message, with high probability (that is, the probability goes to 1 as \( N \) tends to infinity).

First, we analyze the naive algorithm. In this algorithm, each informed node sends a message at every time unit to a node chosen randomly with uniform distribution.

**Theorem 1.1.** For the naive algorithm, with high probability, all active nodes will receive the message after \( (1 + o(1)) \left( \frac{1}{\log(1+p)} + \frac{1}{p} \right) \log N \) time units.

Second, we analyze the cyclic algorithm, which was suggested by Amnon Barak \[2\]. This algorithm begins like the naive algorithm, with messages sent at random, for \( \frac{(1+o(1)) \log N}{\log(1+p)} \) time units. At this point, we can be sure that most active nodes have received the message, and each node starts sending messages to the nodes next to him in cyclic order. More precisely, if we denote the nodes by \( \{1, \ldots, N\} \) then node \( i \) sends a message to node \( i + 1, i + 2, \ldots \) etc. modulo \( N \).

**Theorem 1.2.** For the cyclic algorithm, with high probability, all active nodes will receive the message after \( (1 + o(1)) \left( \frac{1}{\log(1+p)} + \frac{1}{-\log(1-p)} \right) \log N \) time units.

Third, we introduce the improved cyclic algorithm. This algorithm, like the cyclic algorithm, also begins with messages sent at random for \( \frac{(1+o(1)) \log N}{\log(1+p)} \) time units. Then, we divide the network into disjoint segments, each consisting of \( \ell = \sqrt{\log N} \) consecutive nodes (in the cyclic order). A segment is good if it has at least \( \frac{\ell p}{2} \) active nodes, at least one of which is informed, and bad otherwise. Now, each node informs all nodes in its segment. If a segment is bad, then the nodes in it stop transmitting. If a segment is good, then all active nodes in the segment (which are now informed) begin to inform the nodes in the next segments (in the cyclic order) and then the segment after it and so on. This takes \( \lceil 2/p \rceil \) time units per segment, and, since bad segments are rare, we only need to do this for a short time, so this part of the algorithm takes only \( o(\log N) \) time units.

**Theorem 1.3.** For the improved cyclic algorithm, with high probability, all active nodes will receive the message after \( (1 + o(1)) \left( \frac{\log N}{\log(1+p)} \right) \) time units.

Finally, we prove that the improved cyclic algorithm is asymptotically optimal.
Theorem 1.4. For any algorithm, with high probability, the number of time units required for all the active nodes to be informed is at least \((1+o(1))(\log N / \log(1+p))\).

Note that in our model, nodes are active independently with probability \(p\). Our results can be applied to the case where exactly \(K\) random nodes are active by comparing to the independent model with \(p\) chosen to be slightly less then \(K/N\), so that the number of active nodes will be less then \(K\) with high probability (see Lemma 3.1). If the \(K\) active nodes are chosen by an adversary, the naive algorithm is not affected, but for the cyclic and improved cyclic algorithms we need to apply a random permutation to the labels of the vertices and this information needs to be transmitted along with the message.

2 Related Works

The topic of broadcasting information to all nodes of a network has been extensively studied for many different models. For a survey of the different models, see Pelc [16]. Gasieniec and Pelc [11] gave an algorithm working in \(O(\log^2 n)\) (under slightly different assumptions then ours), Diks and Pelc [7] improved this to \(O(\log n)\) (see also [8]). Results for some variation on the model can be found in [4, 6, 3, 15, 5]. In all of these the analysis is up to a constant, whereas we determine the optimal running time up to \(1 + o(1)\).

For examples of real world systems using such broadcasting algorithms, see [1, 13].

Frieze and Grimmett [10] studied the running time of the naive algorithm when there are no faults \((p = 1)\). Their result was further refined by Pittel [17]. Our Theorem 1.1 generalizes these results to all \(0 \leq p \leq 1\). Doerr, Huber and Levavi [9] analyzed the naive algorithm in the case of faulty links, instead of nodes, getting the same running time as in our Thoerem 1.1.

3 Preliminaries

Suppose we have a network with \(N\) nodes, with every node connected directly to the rest of the nodes, but only some of them are active. We assume that each node is active with some fixed probability \(0 < p < 1\), independently from the rest of the nodes. Let \(n\) be the number of active nodes.
Lemma 3.1. With high probability,
\[ pN - N^{2/3} < n < pN + N^{2/3} \] (1)

Proof. \( n \sim \text{Bin}(N, p) \), so \( E[n] = Np \) and \( \text{Var}(n) = N \cdot p \cdot (1 - p) \). By Chebyshev’s inequality,
\[
P(|n - pN| > N^{2/3}) = P(|n - E[n]| > N^{2/3}) \leq \frac{N \cdot p \cdot (1 - p)}{N^{4/3}} \xrightarrow{N \to \infty} 0.
\]

Our goal is to disseminate a piece of information to all active nodes. When \( t = 0 \), only node 0 is informed (we assume that it is active). At every time step \( t \in \mathbb{N} \), each informed node may choose one other node and send a message to it. If the other node is active, then it becomes informed and from time \( t + 1 \) onwards it may also send messages and inform other nodes. The nodes do not know apriori which nodes are active, although this kind of information can be sent from node to node along with the piece of information, at no additional cost.

Let \( k_t \) be the number of informed nodes at time \( t \), and \( T_x \) to be the first time when \( k_t \geq x \). We are interested in the asymptotic behaviour of \( T_n \), under different algorithms. All the running times we find are logarithmic in \( N \), so our results are the form \( T_n = (1 + o(1))C(p) \log(N) \) with high probability. That is,
\[
P\left( (C(p) - \epsilon) \log(N) \leq T_n \leq (C(p) + \epsilon) \log(N) \right) \to 1
\]
as \( N \to \infty \), for any fixed \( \epsilon \) and \( p \).

4 The Naive Algorithm

In the naive algorithm each node sends the message to an independent, uniformly random node.

Fix some \( 0 < \epsilon < \frac{1}{2} \). In order to analyze the running time of the algorithm, we shall divide it into three stages:

- Stage 1: From \( t = 0 \) where only one node is informed, until \( \epsilon pN \) nodes are informed, i.e. from time 0 to \( T_{\epsilon pN} \).
• Stage 2: From time $T_{\epsilon pN}$ to $T_{(1-\epsilon)pN}$.

• Stage 3: From time $T_{(1-\epsilon)pN}$ to $T_n$.

We now analyze the time it takes for the naive algorithm to conclude each stage.

**Lemma 4.1.** For any $\delta > 0$, there exists $m_1 \in \mathbb{N}$ that does not depend on $N$, such that

$$\Pr(k_{t_1 + m_1} < \epsilon pN) < \delta.$$  

**Proof.** Define $Z_t$ to be a Galton-Watson branching process in which every node has two children with probability $p' = p(1 - \epsilon)$, and one child with probability $1 - p'$. The expected number of children is $1 + p'$. As long as $k_t \leq \epsilon pN$, $Z_t$ is dominated by $k_t$ since we can couple them together, and for each node in $k_t$, the probability of informing an active uninformed node at this point is larger than the probability of each node in $Z_t$ having two children. Therefore, it is enough to show that there exists such a $m \in \mathbb{N}$ for $Z_t$, i.e. $m$ such that $\Pr(Z_{t_1 + m} < \epsilon pN) < \delta$.

Define $W_q = \frac{Z_q}{(1+p')^q}$. It is known from branching process theory (see [14] and [12] p.14) that $W_q$ is a martingale, and $W_q$ converges almost surely to some random variable $W$. Since the offspring distribution has finite support and the probability for no offsprings is zero, by Kesten-Stigum theorem we get that $\mathbb{E}[W] = 1$ and that $\Pr(W = 0) = 0$. $W_q$ is a martingale, so by Doob’s optional stopping theorem we get that for any $q$,

$$\Pr(W > 2W_q \mid W_q) \leq \frac{1}{2}.$$  

For $\ell, q \in \mathbb{N}$, define $A$ to be the event $W_q < \frac{1}{(1+p')^\ell}$, so $\Pr(W \leq \frac{2}{(1+p')^\ell} \mid A) \geq \frac{1}{2}$. Therefore,

$$\frac{1}{2} \leq \Pr(W \leq \frac{2}{(1+p')^\ell} \mid A) = \frac{\Pr(W \leq \frac{2}{(1+p')^\ell} \cap A)}{\Pr(A)} \leq \frac{\Pr(W \leq \frac{2}{(1+p')^\ell})}{\Pr(A)},$$

so

$$\Pr(A) \leq 2\Pr(W \leq \frac{2}{(1+p')^\ell}).$$
Since \( P(W = 0) = 0 \), there exists \( \ell \in \mathbb{N} \) such that \( 2P(W \leq \frac{2}{(1+p')^\ell}) < \delta \). Taking \( q = t_1 + \ell \), we get

\[
\mathbb{P}(Z_{t_1+\ell} < (1+p')^{t_1}) = \mathbb{P}\left(Z_q < \frac{1}{(1+p')^\ell}\right) = \mathbb{P}\left(W_q < \frac{1}{(1+p')^\ell}\right)
\]

and since \((1+p')^{t_1} = \varepsilon pN\) we get \( \mathbb{P}(k_{t_1+\ell} < \varepsilon pN) < \delta \).

We proved an upper bound of roughly \( \log N / \log(1+p') \) on the time it takes for stage 1 to conclude. The following lemma establishes a corresponding lower bound for any message sending algorithm, which also yields Theorem 1.4.

**Lemma 4.2.** For any \( 0 < a \leq 1 \) and for any algorithm, the probability that after \( \frac{\log N}{\log(1+p')} - K \) steps there are \( apN \) informed nodes is at most \( \frac{1}{ap(1+p')^K} \).

**Proof.** Let \( 0 < a < 1 \). Let us define a new model in which each node knows which nodes already received a message and which didn’t, but not which nodes are active. Furthermore, assume that all informed nodes can coordinate their message sending. In this model, clearly the optimal algorithm would be for each node to send a message to some new node and to make sure no two nodes send messages to the same node. Hence, the number of informed nodes in the optimal algorithm under this model, \( Z_t \), is a branching process that has two children with probability \( p \), and one child with probability \( 1-p \).

Clearly, the new model dominates the old so we can couple \( Z_t \) and \( k_t \) such that \( Z_t \geq k_t \) for every \( t \). Taking \( t = \frac{\log N}{\log(1+p')} - K \) and using Markov’s inequality, we get

\[
\mathbb{P}(k_{t_0} \geq apN) \leq \mathbb{P}(Z_{t_0} \geq apN) \leq \frac{\mathbb{E}[Z_{t_0}]}{apN} = \frac{(1+p)^{\log N}}{apN} \leq \frac{1}{ap(1+p)^K}.
\]

**Proof of Theorem 1.4.** By Lemma 4.2, if we take \( t = \frac{\log N}{\log(1+p')} - K(N) \) where \( K(N) \) is any function which is \( o(\log(N)) \) and \( \omega(1) \) we get that with high probability, the algorithm does not conclude before time \( t \).

Let us now continue with analyzing the stages of the naive algorithm.
Lemma 4.3. For every $\delta > 0$ and $0 < c_1 < c_2 < p$ there exists $m \in \mathbb{N}$ such that

$$\mathbb{P}(T_{c_2N} - T_{c_1N} > m) < \delta.$$ 

Proof. Define $Z_t$ to be a branching process that with probability $p' = p - c_2$ has two children, and $1 - p'$ to have one, so that the expected number of children is $1 + p'$. In order for $Z_t$ to be compatible with $k_t$, we start $Z_t$ at time $T_{c_1N}$ and with value $Z_{T_{c_1N}} = cN$. As long as $k_t < c_2N$, by coupling, $Z_t$ is dominated by $k_t$, so it is enough to show that there exists $m \in \mathbb{N}$ such that $\mathbb{P}(Z_{T_{c_1N} + m} < c_2N) < \delta$.

Defining $W_q = \frac{Z_q}{(1 + p')^q}$, the conditional expectation $\mathbb{E}[Z_{T_{c_1N} + \ell} | Z_{T_{c_1N}}] = c_1N \cdot (1 + p')^\ell$ will exceed $c_2N$ when $(1 + p')^\ell = \frac{c_2}{c_1} \iff \ell = \frac{\log(c_2)}{\log(1 + p')}$. By the same argument as in Lemma 4.1 there exists $m' \in \mathbb{N}$ such that $\mathbb{P}(Z_{T_{c_1N} + \ell + m'} < c_2N) < \delta$, so taking $m = \ell + m'$ we get the wanted result. \hfill \Box

In particular, we see that for any $\delta > 0$, we can choose $m_2$ such that $\mathbb{P}(T_{(1-\epsilon)pN} - T_{\epsilon pN} > m_2) < \delta$.

Define $t_3 = \frac{\log N}{p(1-\epsilon)}$.

Lemma 4.4. For $\delta > 0$, there exists $m_3$ such that

$$\mathbb{P}(T_n - T_{(1-\epsilon)pN} > t_3 + m_3) < \delta.$$ 

Proof. Define $X_m$ to be the number of uninformed nodes after $m$ steps in stage three. By lemma 3.1 we have $X_0 < 2\epsilon pN$ with high probability.

In the third stage, $k_t > (1 - \epsilon)pN$. For a node not informed yet, the probability of not being informed by a specific node is $(1 - \frac{1}{N})$, so the probability of not being informed by any of the informed nodes at a certain step is smaller than

$$(1 - \frac{1}{N})^{(1-\epsilon)pN} \leq e^{-p(1-\epsilon)} \quad (2)$$

and after $m$ steps in stage 3, the probability of any specific node not being informed is smaller than $e^{-p(1-\epsilon)m}$, so $\mathbb{E}[X_m] \leq 2\epsilon p N e^{-p(1-\epsilon)m}$. Therefore, using Markov's inequality, the probability that there is at least one uninformed node is bounded by

$$\mathbb{P}(X_m \geq 1) \leq \mathbb{E}[X_m],$$

7
and $\mathbb{E}[X_m] < \delta$ will hold when

$$2\epsilon p Ne^{-p(1-\epsilon)m} < \delta \iff m > \frac{\log N}{p(1-\epsilon)} + \frac{\log \left(\frac{2p\epsilon}{\delta}\right)}{p(1-\epsilon)}.$$ 

Taking $m_3 = \frac{\log \left(\frac{2p\epsilon}{\delta}\right)}{p(1-\epsilon)}$, we get the desired result. \qed

Denote $t'_3 = \frac{\log N}{p}$. 

**Lemma 4.5.** For $\delta > 0$, there exists $m$ that does not depend on $N$, such that

$$\mathbb{P}(T_n - T_{(1-\epsilon)pN < t'_3 - m}) < \delta.$$ 

**Proof.** We shall describe a different model which dominates our model and the lemma holds for it. In the new model, all active nodes send messages, even the uninformed nodes. If an uninformed node receives a message from an uninformed node, it becomes informed. Start with $Y_0 = \frac{\epsilon}{2} pN$ uninformed nodes and let $Y_k$ to be the number of uninformed nodes after $k$ steps of this model. Obviously, we can couple this new model with our original model so that the number of uninformed nodes in the original model after $k$ steps of the third stage is at least $Y_k$.

The expectation of $Y_k$ is

$$\mathbb{E}[Y_k] = \frac{\epsilon}{2} pN \left(1 - \frac{1}{N}\right)^{pNk} = \frac{\epsilon}{2} pN e^{-(1+o(1))pk}.$$ 

Let us calculate $Var(Y_k)$. Denote by $B$ the set of uninformed nodes at the onset (so $|B| = \frac{\epsilon}{2} pN$), and $Y_{i,k}$ to be 1 if the $i$-th node is uninformed after $k$ steps of the new model, so that $Y_k = \sum_{i \in B} Y_{i,k}$.

$$Var(Y_k) = \mathbb{E}[Y_k^2] - \mathbb{E}[Y_k]^2 = \sum_{i \in B} \mathbb{E}[Y_{i,k}^2] + \sum_{i \neq j} \mathbb{E}[Y_{i,k}Y_{j,k}] - \mathbb{E}[Y_k]^2.$$ 

Observe that $\mathbb{P}(Y_{j,k} = 1|Y_{i,k} = 1) \leq \mathbb{P}(Y_{j,k} = 1)$, and therefore

$$\mathbb{E}[Y_{i,k}Y_{j,k}] = \mathbb{P}(Y_{i,k} = 1 \cap Y_{j,k} = 1)
= \mathbb{P}(Y_{i,k} = 1) \mathbb{P}(Y_{j,k} = 1|Y_{i,k} = 1) \leq \mathbb{P}(Y_{i,k} = 1)^2.$$ 

8
Now, since \( \mathbb{E}[Y_k] = \frac{\epsilon p N}{2} \cdot \mathbb{P}(Y_{i,k} = 1) \), we get

\[
\sum_{i \neq j} \mathbb{E}[Y_{i,k}Y_{j,k}] \leq \left( \frac{\epsilon p N}{2} \right)^2 \cdot \mathbb{P}(Y_{i,k})^2 = \mathbb{E}[Y_k]^2,
\]

and therefore

\[
\text{Var}(Y_k) \leq \sum_{i \in B} \mathbb{E}[Y_{i,k}^2] = \mathbb{E}[Y_k].
\]

By Chebyshev’s inequality we now have

\[
\mathbb{P}(Y_k = 0) \leq \mathbb{P}(|Y_k - \mathbb{E}[Y_k]| \geq \mathbb{E}[Y_k]) \leq \frac{\text{Var}(Y_k)}{\mathbb{E}[Y_k]^2} \leq \frac{1}{\mathbb{E}[Y_k]}.
\]

Therefore, as long as \( \mathbb{E}[Y_k] \geq \frac{1}{\delta} \), the probability of finishing is smaller than \( \delta \). Taking \( k = \log(\frac{\delta p N/2}{p}) \) yields \( \mathbb{E}[Y_k] \geq \frac{1}{\delta} \), so taking \( 0 < m = -\frac{\log(\delta p/2)}{p} \) we get the desired result. \( \square \)

We can now prove Theorem 1.1.

**Proof of Theorem 1.1.** Observe that \( T_n = T_n - T_{(1-\epsilon)pN} + T_{(1-\epsilon)pN} - T_{\epsilon pN} + T_{\epsilon pN} \). By Lemma 4.1 we know that there exists \( m_1 \in \mathbb{N} \) such that \( T_{\epsilon pN} \leq \frac{\log N}{\log(1+p(1-\epsilon))} + m_1 \) with probability larger than \( 1 - \delta \). By Lemma 4.3 we know that there exists \( m_2 \in \mathbb{N} \) such that with probability larger than \( 1 - \delta \), \( T_{(1-\epsilon)pN} - T_{\epsilon pN} > m_2 \). By Lemma 4.4 we know that there exists \( m_3 \in \mathbb{N} \) such that with probability larger than \( 1 - \delta \), \( T_n - T_{(1-\epsilon)pN} < \frac{\log N}{p} + m_3 \). Taking \( N \to \infty \), and \( \epsilon, \delta \to 0 \) slowly enough such that \( m_1, m_2, m_3 = o(\log N) \), we get that for the naive algorithm, with high probability, all active nodes will be informed by time \( (1 + o(1)) \left( \frac{1}{\log(1+p)} + \frac{1}{p} \right) \log N \).

By Lemma 4.2 we know there exists some \( m_4 \) such that is stage 1 is not done before \( \frac{\log N}{\log(1+p)} - m_4 \), with probability at least \( 1 - \delta \). By Lemma 4.5 we know that there exists \( m_5 \) such that with probability larger than \( 1 - \delta \), stage 3 is not done before \( \frac{\log N}{p} - m_5 \). Again, taking \( N \to \infty \), and \( \epsilon, \delta \to 0 \) slowly enough such that \( m_4 \) and \( m_5 \) are \( o(\log N) \), we get that the naive algorithm, with high probability, will not be done before time \( (1 + o(1)) \left( \frac{1}{\log(1+p)} + \frac{1}{p} \right) \log N \) \( \square \)

## 5 The Cyclic Algorithm

As Lemmas 4.1, 4.2 and 4.3 show, the first and second stages of the naive algorithm are optimal. The cyclic algorithm consists of two phases (not to be confused with the three stages of analysis).
The first phase consists of sending messages randomly, as in the naive algorithm, until some time \((1 + o(1)) \frac{\log N}{\log(1 + p)}\) which guarantee (by Lemmas 4.1 and 4.3) that \((1 - \epsilon)pN\) nodes are informed, with high probability, for some fixed \(\epsilon > 0\). In order for all the informed nodes to know when the first phase has finished, each nodes includes the time (as measured from the start of the algorithm) in the information it sends.

In the second phase, nodes are sending messages in a cyclic order: Mark the nodes \(\{1, \ldots, N\}\). If node \(i\) is informed at the beginning of the second phase, it will send messages to node \(i + 1 \mod N\), then \(i + 2 \mod N\), and so on.

Denote \(t_5 = -\frac{\log N}{\log(1 - p(1 - \epsilon))}\). Let \(T'_n\) be the number of steps from the start of the second phase until all nodes are informed.

**Lemma 5.1.** For \(\delta > 0\), there exists \(m\) such that \(\Pr(T'_n > t_5 + m) < \delta\).

**Proof.** Observe that at the beginning of the second phase, there are, with high probability, at least \((1 - \epsilon)pN\) informed nodes. Since the first phase is invariant under all permutations, the probability that any specific \(k\) nodes are all uninformed is bounded by

\[
\binom{N-k}{(1-\epsilon)pN} \leq (1 - (1 - \epsilon)p)^k.
\]

Hence, if we take \(k = \frac{\log \delta - \log N}{\log(1 - p(1 - \epsilon))}\) and using union bound, we get that the probability that there are \(k\) consecutive (in the cyclic order) uninformed nodes is at most \(\delta\). In other words, the probability of the second phase not concluding after \(k\) steps is at most \(\delta\). Taking \(m = \frac{\log \delta}{\log(1 - p(1 - \epsilon))}\) we get the desired result.

We can now prove Theorem 1.2.

**Proof of Theorem 1.2.** As noted at the beginning of the section, by the end of the first phase, which takes \((1 + o(1)) \frac{\log N}{\log(1 + p)}\) time steps, with high probability there are \((1 - \epsilon)pN\) informed nodes, for some \(\epsilon > 0\). By Lemma 5.1, for any \(\delta > 0\), there is some \(m\) such that the probability of having uninformed nodes at time \(-\frac{\log N}{\log(1 - p(1 - \epsilon))} + m\) is at most \(\delta\). Taking \(N \rightarrow \infty\), and \(\epsilon, \delta \rightarrow 0\), slowly enough such that \(m = o(\log N)\) yields the desired result.

The next theorem show that this algorithm is indeed an improvement over the naive algorithm.
**Theorem 5.2.** For every $0 < p < 1$, the cyclic algorithm’s running time is asymptotically better than the naive algorithm’s running time.

**Proof.** We need to show that $\frac{1}{p} > \frac{1}{-\log(1-p)}$ for $0 < p < 1$, which happens if and only if $p + \log(1-p) < 0$. Define

$$f(p) = p + \log(1-p).$$

Then $f(0) = 0$ and the derivative

$$f'(p) = 1 - \frac{1}{1-p}$$

is negative for any $0 < p < 1$, so $f(p) < 0$ for any $0 < p < 1$. □

### 6 The Improved Cyclic Algorithm

As noted earlier, the first phase of the cyclic algorithm is optimal. The improved cyclic algorithm will have the same first phase and a second phase, described below, which will take only $o(\log(N))$ steps to inform all nodes. Notice that at the end of the first phase, there are, with high probability, less then $(1-(1-\epsilon)p)$ uninformed nodes and they are invariant under all permutations of the nodes. Consider a model where, each active node is informed with probability $(1-2\epsilon)p$, independently. Call this the independent model. Comparing to the situation at the end of the first phase, in the independent model there will be more uninformed nodes, with high probability, and they are also invariant under all permutations. Since our second phase is going to be monotone, that is, changing a node from uninformed to informed will not cause the second phase to fail, it is enough to prove that it works with high probability under the independent model.

Let $\ell(N) = \sqrt{\log N}$ (in fact, any function which is $o(\log N)$ and $\omega(1)$ will do here). Group the nodes $1, 2, \ldots, N$ into $N/\ell$ nonoverlapping segments of $\ell$ contiguous nodes each. Define a segment to be *good* if it has at least $\ell(1-2\epsilon)p/2$ informed nodes at the end of the first phase, and *bad* otherwise. Let $q(N)$ be the probability that a segment is good.

**Lemma 6.1.** Under the independent model, $q(N) \to 1$ as $N \to \infty.$
Proof. Under the independent model, $X$, the number of informed nodes in a segment is Binomial with parameters $\ell$ and $p' = (1 - 2\epsilon)p$. The expectation is $E[X] = \ell p'$ and the variance is $Var(X) = \ell p'(1 - p')$. By Chebyshev inequality we have

$$\Pr(X < \frac{\ell p'}{2} \leq \Pr(|X - \ell p'| > \frac{\ell p'}{2}) \leq \frac{\ell p'(1 - p')}{(\ell p')^2} < \frac{1}{\ell p'}$$

and this bound tends to 0, since $\ell \to \infty$.

At the beginning of the second phase, each informed node informs the other nodes in its segment. This takes at most $\ell$ (which is $o(\log N)$) steps, after which each node in the segment knows which other nodes are informed and which are not and specifically, whether the segment is good or bad. If the segment is bad, then the nodes stop sending messages. If the segment is good, then all informed nodes begin to inform the node of the next segment (in the cyclic order), making sure that different nodes informs different nodes in the next segment. This takes only a constant number of steps, specifically, at most $\ell / (k/2) \leq \frac{2}{p}$ steps. After that, they start informing the next segment, and so on.

**Lemma 6.2.** With high probability $T_n - T'(1 - \epsilon)pN$ is $o(\log N)$.

Proof. The first part of the second phase takes $\ell$ steps which is $o(\log N)$. After that, the time it takes to finish the second phase is at most $\frac{3}{p}$ times the length of the longest sequence of bad segments. Under the independent model, the probability that there are $k$ consecutive bad segments at a specific location is $(1 - q)^k$. Using union bound, the probability of getting such a sequence anywhere is at most $N(1 - q)^{k}$. Plugging in $k = \frac{2\log N}{\log(1 - q)}$ we get that this probability is bounded by $\frac{1}{N}$. By Lemma 6.1, $q \to 1$, so our choice of $k$ is $o(\log N)$.

**Proof of Theorem 1.3.** By Lemmas 4.1 and 4.3, the first phase takes $(1 + o(1)) \frac{\log N}{\log(1 + \epsilon)}$, with high probability. By Lemma 6.2, the second phase takes $o(\log N)$. By Theorem 1.4, this is asymptotically optimal.
Acknowledgments

We thank Amnon Barak for introducing us to the subject and to the cyclic algorithm.

References

[1] A. Barak, Z. Drezner, E. Levy, M. Lieber and A. Shiloh. Resilient gossip algorithms for collecting online management information in exascale clusters. Concurrency and Computation: Practice and Experience, Vol. 27(17):4797-4818, Dec. 2015.
[2] A. Barak, private discussion.
[3] P. Berman, K. Diks and A. Pelc, Reliable broadcasting in logarithmic time with Byzantine link failures, Journal of Algorithms 22 (1997), 199-211.
[4] B.S. Chlebus, K. Diks and A. Pelc, Sparse networks supporting efficient reliable broadcasting, Nordic Journal of Computing, Vol 1, No. 3 (1994). pp. 332-345.
[5] K. Diks and A. Pelc, Efficient gossiping by packets in networks with random faults, SIAM Journal on Discrete Mathematics 9 (1996), pp 8-17.
[6] K. Diks and A. Pelc, Almost safe gossiping in bounded degree networks, SIAM Journal on Discrete Mathematics 5 (1992), pp. 338-344.
[7] K. Diks and A. Pelc, Optimal Adaptive Broadcasting with a Bounded Fraction of Faulty Nodes, Algorithmica 28 (2000) Issue 1, pp 3750
[8] B. Doerr, C. Doerr, S. Moran and S. Moran Simple and optimal randomized fault-tolerant rumor spreading, Distributed Computing 29 (2016) Issue 2, pp 89104.
[9] B. Doerr, A. Huber and A. Levavi2Strong robustness of randomized rumor spreading protocols, Discrete Applied Mathematics 161 (2013), Issue 6, pp 778-793
[10] A. M. Frieze and G. R. Grimmett, The shortest-path problem for graphs with random arc-lengths, Discrete Applied Mathematics 10 (1985), 57-77.
[11] L. Gasieniec and A. Pelc, *Adaptive broadcasting with faulty nodes*, Parallel Computing 22 (1996), 903-912

[12] T.H. Harris, *The Theory of Branching processes*, Dover Publications, Inc., Mineola, NY, 2002.

[13] E. Levy, A. Barak, A. Shiloh, M. Lieber, C. Weinhold and H. Hertig. *Overhead of a decentralized gossip algorithm on the performance of HPC applications* Proc. Intr. Workshop on Runtime and Operating Systems for Supercomputers (ROSS), Munich, June 2014.

[14] R. Lyons, R. Pemantle and Y. Peres, *Conceptual proofs of LlogL Criteria for mean behavior of branching processes*, Ann. Probab. 23 (1995), no. 3, 1125-1138.

[15] P. Panaite and A. Pelc, *Optimal broadcasting in faulty trees*, Journal of Parallel and Distributed Computing 60 (2000), 566-584.

[16] A. Pelc, *Fault-tolerant broadcasting and gossiping in communication networks*, Networks 28 (1996), 143-156.

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