

**CS599: Structure and Dynamics of Networked Information (Spring 2005)**  
**04/20/2005: Gossip Algorithms**  
**Scribes: Ramakrishna Gummadi and Affan Syed**

In this lecture, we consider how to distribute a piece of information, like a rumor, within a network. Naturally, the ability to simply spread one piece of information lies of the core of other, more complicated, network protocols. A first and common approach for the task is to build a tree on the nodes, and forward the message along edges of the tree. So long as the height of the tree is logarithmic, and the degrees bounded by a constant (e.g., for a complete binary tree), the message will reach all nodes in  $O(\log n)$  steps. The best completion time (time by which the last node receives the message) is achieved by a binomial tree.

The problem with any tree is that it is not fault-tolerant. If one node fails, then none of the nodes in the subtree rooted at it will receive the message. In order to achieve more fault-tolerance, we will need more redundancy in messages sent. One could make the graph 2-connected, or more highly connected. A better alternative perhaps is to use epidemic algorithms (also called gossip algorithms), which mimic the behavior of epidemics, in that they use information exchanges between random node pairs. Their particular advantage lies in their simplicity and inherent fault-tolerance; they pay for it with high message redundancy.

In all versions of *gossip protocols* studied here, we assume that there are synchronous rounds, and in each round, each node calls a random other node and exchanges information. These exchanges can be divided into three categories:

1. In *push gossip*, an (informed) sender randomly picks a partner and sends a message.
2. In *pull gossip*, each node picks a random partner, and receives a message from the partner (if it has one).
3. In *push & pull gossip*, each node picks a random partner, and both forwards and receives a message.

Naturally, push & pull gossip makes most sense when multiple messages are being exchanged, as otherwise, no informed node ever needs to pull, and no uninformed node can ever push. For instance, exchanging information in both directions is useful in synchronizing a replicated distributed database periodically.

In deciding whether to use gossip as a primitive, an important consideration is its speed: how quickly do all nodes receive the message? We expect the time to be roughly  $O(\log n)$ , as the number of nodes having the message roughly doubles until most nodes already have the message.

**Proposition 1** *It takes  $O(\log n)$  rounds to get a message across to everyone.*

**Proof.** We first analyze the behavior until more than  $n/3$  nodes have the message. So consider a time  $t$  when at most  $m \leq \frac{n}{3}$  nodes have the message. Then, each of the  $m$  messages sent has probability at least  $2/3$  of reaching a previously uninformed node, so in expectation, at least  $\frac{2m}{3}$  sent messages reach previously uninformed nodes.

Unluckily, this does not quite guarantee that  $2m/3$  new nodes will be informed, as some of these messages will reach the same nodes. We want to upper-bound how frequently this happens. For any given pair of messages, the probability that they both go to the same uninformed node is  $O(\frac{1}{n-m})$ . As there are at most  $\binom{m}{2}$  such pairs, the expected number of such collisions is at most  $\frac{m^2}{2(n-m)} \leq \frac{m^2}{2} \cdot \frac{1}{2m} = \frac{m}{4}$ . Thus, the expected number of newly infected nodes is at least  $\frac{2m}{3} - \frac{m}{4} = \frac{5m}{12}$ .

Let the random variable  $X_t$  be the number of informed nodes at time  $t$ . By our arguments above,  $X_0 = 1$ , and  $E[X_{t+1} | X_t = m \leq \frac{n}{3}] \geq \frac{17}{12}m$ . By induction,  $E[X_t] \geq (\frac{17}{12})^t$ . We can now apply Markov's Inequality to show that with high probability, after  $O(\log n)$  rounds, at least  $\frac{n}{3}$  nodes are active.

Once  $\frac{n}{3}$  nodes are active, any other node  $v$  is *not* called by an informed node with probability at most  $(1 - \frac{1}{n})^{\frac{n}{3}} \leq e^{-\frac{1}{3}} = e^{-1/3}$ . So after  $t = 6 \log n$  independent rounds of this process, the probability that a particular node  $v$  is still uninformed is at most  $(e^{-1/3})^{6 \log n} = n^{-2}$ . By a Union Bound over all  $n$  nodes, the probability that all nodes get the message in  $O(\log n)$  rounds is then at least  $1 - \frac{1}{n}$ . ■

From the point at which  $n/3 = \Omega(n)$  nodes have the message until the point when *all* nodes have the message, there are  $\Omega(\log n)$  rounds, and in each such round, each informed node sends the message to some other node, for a total of  $\Omega(n \log n)$  messages sent. On the other hand, using any tree would only require sending  $O(n)$  messages. Thus, gossip achieves its fault-tolerance by using  $\Omega(\log n)$  as many messages as necessary.

Naturally, one wonders whether such an amount of overhead is inherent in any fault-tolerant approach, or in any gossip-based approach. Karp et al. [2] give a partial answer to this question. They show that any protocol based on uniform gossip must send  $\omega(n)$  messages, and, under additional restrictions on the protocol,  $\Omega(n \log \log n)$  messages. Conversely, they analyze more carefully the “push-pull” version of gossip, and show that if the terminating point is chosen carefully, only  $O(n \log \log n)$  messages are sent.

**Theorem 2** *If push-pull is run for  $\log_3 n + O(\log \log n)$  rounds, then, w.h.p., all nodes have the message, and, in addition, the total number of messages is  $O(n \log \log n)$ .*

Here, we begin the proof, which will be completed in the next lecture. Solely for the purposes of analysis, we divide the algorithm into four phases.

**Startup phase:** This phase lasts until  $\log^4 n$  nodes have the message with high probability (probability at least  $1 - n^{-c}$  for some  $c$ ). We observe that within  $O(1)$  rounds, the number of informed nodes doubles with high probability (because each informed node will have called a previously uninformed one), and so the entire phase takes  $O(\log \log n)$  rounds for the goal of achieving  $\log^4 n$  informed nodes.

**Exponential growth phase:** This phase lasts until  $\frac{n}{\log n}$  nodes are informed. We write  $S_t$  for the number of nodes having the message at time  $t-1$ , and  $m_t$  the number of messages that were sent in round  $t$ . Because in expectation, each informed node calls one node, and is called by one, we have that  $E[m_t] = 2S_{t-1}$ .

In fact,  $m_t$  is sharply concentrated, i.e., unlikely to deviate far from its expectation. To see this, if we let  $X_{uv} = \begin{cases} 1 & \text{if node } u \text{ calls } v \text{ in round } t \\ 0 & \text{otherwise} \end{cases}$ . Then,  $m_t = \sum_{u \in S_{t-1}} \sum_v (X_{uv} + X_{vu})$ . Because  $m_t$  is a sum of negatively dependent 0–1 variables, Chernoff Bounds show that  $m_t$  is sharply concentrated around its mean.

The rest of the proof is given in the next lecture notes.

**Theorem 3 (Chernoff Bounds [1, 3])** *If  $X_1, X_2, \dots, X_n$  are independent 0–1 random variables, with  $X = \sum_i X_i$ ,  $\mu = E[X] = \sum_i \text{Prob}[X_i = 1]$ , and  $\delta > 0$ , then*

$$\text{Prob}[X > (1 + \delta)\mu] < \left( \frac{e^\delta}{(1 + \delta)^{(1 + \delta)}} \right)^\mu.$$

*Similarly,*

$$\text{Prob}[X < (1 - \delta)\mu] < e^{-\frac{\delta^2}{2}\mu}$$

These bounds extend to sums of negatively correlated 0–1 variables.

## References

- [1] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Math. Statistics*, 23:493–509, 1952.
- [2] R. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. Randomized rumor spreading. In *Proc. 41st IEEE Symp. on Foundations of Computer Science*, pages 565–574, 2000.
- [3] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1990.