

1 Introduction

This lecture notes is based on the presentation of the paper “On the Time-Complexity of Broadcast in Multi-Hop Radio Networks: An Exponential Gap Between Determinism and Randomization” (Bar-Yehuda, Goldreich and Itai) by the instructor.

2 Problem Formulation and Network Model

Broadcasting is the problem of propagating a message from a source to all nodes in a network. We assume that the network is a connected one. The problem is an easy one if we know what the network layout is. Here, we assume an unknown topology. A node knows only its own identifier and the identifiers of its neighbors.

This paper makes the following strong assumptions about what happens when multiple neighbors of a node transmit together and therefore a collision takes place:

- Receiver does not receive any message.
- Receiver cannot distinguish collision from the case when none of the neighbors make any attempt to transmit.

Kowalski and Pelc (2004) found an error in the above-mentioned model of Bar-Yehuda et al. Bar-Yehuda claimed that for any broadcast protocol there is a network G_S - which is a graph of radius 2 with $n + 2$ nodes (node 0 being the source and node $n + 1$ the sink), with a subset S of $1, 2, \dots, n$, such that all the vertices in $1, 2, \dots, n$ are neighbors of the source and all the elements of S are neighbors of the sink (refer to Figure 1)- on which the protocol works in time $\Omega(n)$. Kowalski-Pelc points out that *under the considered model, the result given by Bar-Yehuda does not even work for oblivious algorithms* (an oblivious algorithm is one in which sets of nodes transmitting in a given step must be fixed in advance). Kowalski-Pelc proposed an algorithm that broadcasts in all such above-mentioned network in time $O(\log n)$.

Bar-Yehuda et al assumed that if there are multiple transmissions in the neighborhood of a node, then nothing arrives to the node. Kowalski-Pelc changed the model to a more general

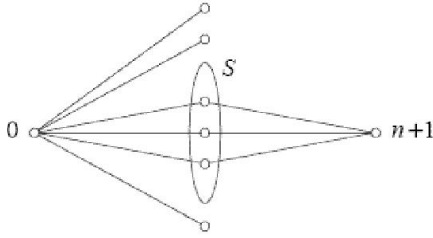


Figure 1: Topology with source and sink

one in the sense that in this latter model, if multiple neighbors transmit together, either no message reaches the receiver, which is same as the case of no transmission, or exactly one message passes through. If the latter happens, the receiver is unaware of the fact that actually more than one nodes had attempted to transmit. Later, we describe the models followed by Bar-Yehuda and Kowalski-Pelc as model A and model B respectively.

The Bar-Yehuda et al paper gives *an upper bound on randomized broadcast*, as well as *a lower bound on deterministic broadcast*. The upper bound result states that for any ϵ , there is a randomized broadcast protocol that takes $O((D + \log \frac{n}{\epsilon}) \log n)$ slots with probability $1 - \epsilon$, where D is the diameter of the network, i.e., the maximum shortest distance between any two nodes in the network.

The lower bound result, as stated before, claims that for any deterministic broadcast protocol, there exists networks that take $\Omega(n)$ time. Since the randomized protocol has a logarithmic upper bound and the deterministic protocol has a linear lower bound, they claim that there exists an exponential gap between deterministic and randomized broadcast protocols.

2.1 Model A

We define the following parameters for the network:

n : number of nodes

D : maximum distance between node 0 (the source) and any other node

N_u : the set of neighbors of node u in the graph, i.e., the set of nodes within the transmission/reception range of u

This model assumes the following network behavior:

- The broadcast operation takes place in synchronous time slots.
- In each slot, each node chooses to be the transmitter or the receiver or remains inactive.

This model assumes the following rules for handling message reception and collision:

- If node u is a receiver in slot k , and exactly one of the nodes in N_u sends in slot k , then u receives.
- If none or more than one nodes in N_u send, then nothing is received at u . The case in which two or more neighbors transmit is indistinguishable from the case in which no neighbor transmits. In both cases, u hears noise, which is always present, even when none in N_u transmits.

2.2 Model B

This model assumes the following rules for handling message reception and collision:

- If node u is a receiver in slot k , and no node in N_u transmits in slot k , then u receives a null message
- If node u is a receiver in slot k , and more than one nodes in N_u transmit in slot k , then the result may be either of the following possibilities:
 - One of these transmissions is received, like the case of a transmission by exactly one neighbor
 - Nothing is received, like the case when no neighbor transmits

The case of multiple transmission is a bad event and its outcome is unpredictable.

An execution under model A is also a valid execution under model B, but the converse does not hold. In model A, the message delivery events are fully determined by the number of neighbors that transmit, whereas in model B in some cases (i.e., multiple transmitters) delivery is decided non-deterministically (i.e., by an adversary).

2.3 Randomized model

This model is basically similar to model A. Additionally, it is assumed that the nodes have an a priori knowledge of a parameter ϵ , and they are supposed to achieve broadcast with probability $1 - \epsilon$. They also know another parameter Δ , which is an upper bound on the maximum node degree.

This model makes use of a randomized sub-protocol named *Decay*. The intuition behind the *Decay* procedure is as follows: a processor receives a message in a certain time-slot if and only if exactly one of its neighbors acts as a transmitter during this time-slot. Thus, in order to guarantee that a message is received, one must coordinate the neighbors so that exactly one of them transmits. Coordinating neighbors by deterministic means is highly inefficient, since the coordination channels are subject to exactly the same difficulties. Thus, instead of trying to achieve deterministic coordination, we turn for help to randomization procedures. Suppose

$d \leq \Delta$ processors compete for a time-slot in which exactly one of them sends a message. Simultaneously, they all start a game of coin flips. At each time-slot, on the average half of the remaining processors remove their candidacy. It can be shown that, with constant probability, before all processors remove their candidacy there exists a time-slot with exactly one candidate. The pseudocode of the algorithm follows, with input parameter $k \geq 1$ and message m :

```

procedure Decay( $k, m$ )
 $coin := 1$ ;
 $counter := k$ ;
while  $coin = 1$  and  $counter > 0$  do
  transmit  $m$ ;
   $counter := counter - 1$ ;
  set  $coin := 0$  or  $coin := 1$  with equal probability
endwhile
end

```

Suppose d nodes execute *Decay* in slot 1. Let $P(k, d)$ be the probability that at some slot $\leq k$, exactly one node transmits.

Let $P(\infty, d) = \lim_{k \rightarrow \infty} P(k, d) =$ Probability that exactly one node *eventually* transmits.

Now, the following two results hold trivially:

$P(k, 0) = 0$ since if no node executes *Decay*, there will be no transmissions.

$P(k, 1) = 1$ since if only one node executes *Decay*, it will definitely transmit.

Theorem 1 *Let u be a node in v . Suppose $d \geq 2$*

(1) $P(\infty, d) \geq \frac{2}{3}$

(2) For $k \geq \lceil \log(d) \rceil$, $P(k, d) > \frac{1}{2}$

Proof: (1) $P(\infty, d) \geq \frac{2}{3}$

To find the probability that up to d remain after one decay step, we break it up into the simpler calculations where each one is the probability that exactly i remain. The probability that up to d remain is then the sum of all i from zero to d .

$$\begin{aligned}
 P(\infty, d) &= \sum_{i=0}^d \binom{d}{i} 2^{-d} P(\infty, i) \\
 &= \binom{d}{0} 2^{-d} P(\infty, 0) + \sum_{i=1}^{d-1} \binom{d}{i} 2^{-d} P(\infty, i) + \binom{d}{d} 2^{-d} P(\infty, d) \\
 \Rightarrow 2^d P(\infty, d) &= \sum_{i=1}^{d-1} \binom{d}{i} P(\infty, i) + \binom{d}{d} P(\infty, d) \\
 \Rightarrow (2^d - 1) P(\infty, d) &= \sum_{i=1}^{d-1} \binom{d}{i} P(\infty, i)
 \end{aligned}$$

Prove $P(\infty, d) \geq \frac{2}{3}$ by induction on d :

Base Case: $d = 2$

$$3P(\infty, 2) = \binom{2}{1} P(\infty, 1) = 2$$

$$\Rightarrow P(\infty, 2) = \frac{2}{3}$$

Inductive Step: Assume $P(\infty, d') \geq \frac{2}{3}$ for $d' < d$

Prove $P(\infty, d) \geq \frac{2}{3}$

$$\begin{aligned} P(\infty, d)(2^d - 1) &= \sum_{i=1}^{d-1} \binom{d}{i} P(\infty, i) \\ &= \binom{d}{1} P(\infty, 1) + \sum_{i=2}^{d-2} \binom{d}{i} P(\infty, i) \text{ noting that } P(\infty, i) \geq \frac{2}{3} \\ &\geq d + \frac{2}{3} \sum_{i=2}^{d-2} \binom{d}{i} \\ &= \sum_{i=2}^{d-2} \binom{d}{i} = 2^d - 1 - 1 - d \\ \text{So RHS} &\geq d + \frac{2}{3}(2^d - 1) - \frac{2}{3}(d + 1) \\ &\geq \frac{2}{3}(2^d - 1) \text{ if } d \geq \frac{2}{3}(d + 1) \text{ for } d \geq 2 \\ &\Rightarrow P(\infty, d) \geq \frac{2}{3} \text{ QED} \end{aligned}$$

■

Proof: (2) For $k \geq \lceil \log(d) \rceil$, $P(k, d) > \frac{1}{2}$

Values up to $d = 5$ hold by inspection

$$d \geq 6 \Rightarrow P(k, d) \geq P(\infty, d) - d2^{-k}$$

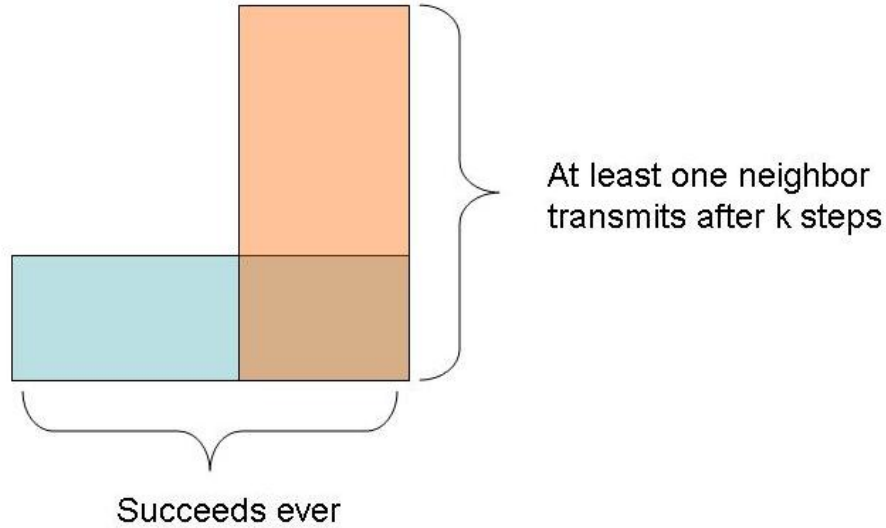
Probability of success in k slots \geq probability of success ever in infinite slots minus probability at least one neighbor transmits after k slots.

■

Note: If node ever gets a message and all neighbors transmit in no more than k slots, then node must get message in k slots. So, probability of success within k slots \geq probability of success ever minus probability at least one neighbor transmits after k slots.

$$P(\text{ succeeds in } k \text{ steps}) \geq P(\text{ succeeds sometime}) - P(\text{ at least one neighbor transmits})$$

The reason this is an inequality is to take into account the case that after k steps all transmissions result in collisions. This is illustrated in the Venn diagram below.



Broadcast

To ensure probability $1 - \epsilon$ given ϵ run $\text{Decay}(k, m)$ t times $t = 2 \lceil \lg(\frac{N}{\epsilon}) \rceil$ each time using $k = 2 \lceil \lg \Delta \rceil$

Outline of algorithm

- nodes group slots into synchronized batches of $2 \lceil \lg \Delta \rceil$ slots.
- each node waits till it receives source message m
- next synchronized batch of slots executes $\text{decay}(k, m)$ for t times.

Lemma All nodes eventually receive message is $\geq 1 - \epsilon$

Proof: Suppose that the broadcast fails, node u did not receive the message, but its neighbor node v did.

$\Pr(\text{of the above}) \leq n \cdot \Pr(\text{a particular node } u \text{ does not receive but a neighbor does})$ This is an upper bound since we ignore possible interdependence

So focusing on a particular u there are at least t attempts by some neighbor v using $\text{Decay}(k, m)$, each time $\text{Decay}(k, m)$ had a probability to succeed $\geq \frac{1}{2}$. However all attempts failed. The probability of failure in t attempts $< \frac{1}{2}^t \leq \frac{1}{2}^{\lg \frac{N}{\epsilon}} = \frac{\epsilon}{N}$

$\Pr(\text{some such node } u \text{ exists}) \leq n \left(\frac{\epsilon}{n}\right) \leq \epsilon$

■

$O(D \log(\Delta) \log(\frac{N}{\epsilon}))$ where D is the number of hops, $\log \Delta$ is the number of slots, and $\log(\frac{N}{\epsilon})$ is the number of decay runs.

A more detailed analysis can reach $O(\log(\Delta)(D + \log(\frac{N}{\epsilon})))$