ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

FS 2015

Prof. R. Wattenhofer
David Stolz

Principles of Distributed Computing
Exercise 12: Sample Solution

1 Determining the Median

As stated in the hint, we start with initializing the nodes to give them IDs from $1, \ldots, n$. Now the node with ID i transmits its token in time slot i. Each node uses two variables to count the number of tokens which were transmitted with a higher or lower number. After the execution each node knows how many tokens were larger/smaller than its own. Thus, the node whose token is the median can simply transmit it afterwards.
Let us briefly analyze the time and space used by this. We know from the lecture that initialization takes $\mathcal{O}(n)$ rounds. The sending of each token afterwards takes exactly n rounds, i.e., does not increase the asymptotic runtime.
Each node needs to store a new ID of up to n, i.e., needs $\mathcal{O}(\log n)$ space. The variables from the algorithm can also be stored by using $\mathcal{O}(\log n)$. After the initialization, we require that each node keeps two counters to count how many numbers were larger/smaller than our own. But since we do not need to store the values, this requires only $\log n$ space.
The correctness follows by the construction of the algorithm. The unique node whose upper and lower counter has the same value, can broadcast it and thus all nodes are aware of it.

2 Maximum

```
1:  while TRUE do
2:     elect leader
3:     leader broadcasts value
4:     if If own value is bigger then
5:        broadcast own value
6:     end if
7:     if no transmitter then
8:        leader has max
9:     else if single transmitter then
10:       transmitter has max
11:    else if own value ≤ leader then
12:       exit
13:    end if
14: end while
```

A round is *good* if at least half the vertices exit in that round. Observe that tt most $log(n)$ good rounds are needed to find the maximum. Since the leaders are chosen randomly and independently in each round, a round is good with probability at least $\frac{1}{2}$. Let the random variable G be the number of good rounds after $4c \cdot \log(n)$ rounds, for $c \geq 1$. The expectation of G is bounded by

$$E[G] \geq \frac{1}{2} 4c \cdot \log(n) = 2c \cdot \log(n) \geq 2 \cdot \log(n)$$

Using the Chernoff bound and setting $\delta = \frac{1}{2}$ we get the following as probability of failure (i.e., not having enough good rounds)

$$\begin{aligned} Pr[G < \log(n)] &= Pr[G < (1-\delta)2 \cdot \log(n)] \\ &\leq Pr[G < (1-\delta)E[G]] \\ &\leq e^{-\frac{\delta^2}{2}E[G]} \\ &= e^{-\frac{1}{4}c\log(n)} \\ &= n^{-\frac{c}{4}} \end{aligned}$$

Additionally, we have a probability of failure of $n^{-c'}$ every time we elect a leader. Note that we look at the probability of succeeding in $4c \cdot \log(n)$ rounds, and we perform one leader election in each of these rounds. We set $c' = \frac{c}{4}$; this is possible, since the notion of w.h.p. allows us to choose the constant c' in the exponent of the success probability.
With the union bound we can derive

$$\begin{aligned} Pr[\text{fail}] &= P\left[\{G < \log(n)\} \cup \bigcup_i \{\text{leader election in round } i \text{ failed}\}\right] && \mid \text{Union Bound} \\ &\leq n^{-c'} + 4c \cdot \log(n) n^{-c'} \\ &= (1 + 4c \cdot \log(n)) n^{-c'} && \mid n > 1, c \geq 1 \\ &\leq 5c \cdot \log(n) n^{-c'} \\ &= 2^{\log(5c)} \cdot \log(n) n^{-c'} \\ &< 2^{\log(5c)\cdot\log(n)} n^{-c'} \\ &= n^{-c' + \log(5c)} \\ &= n^{-\alpha} \end{aligned}$$

where α is independent of n and can be set arbitrarily high by choosing the constant c'.