Principles of Distributed Computing 05/05, 2021

Exercise 10

Lecturer: Mohsen Ghaffari

1 Pipelining

We consider an arbitrary n-node network $G = (V, E)$ with diameter D. Moreover, we work in the CONGEST model of distributed computing where each node has an $O(\log n)$-bit unique identifier and per round, each node can send $O(\log n)$ bits to each of its neighbors.

Exercises

(1a) Suppose that each node $v \in V$ is given k different inputs $x_1(v)$, $x_2(v)$, ..., $x_k(v)$, each being a $\Theta(\log n)$-bit number. The objective is for all nodes to know the outputs $y_i = \min_{v \in V} x_i(v)$, for each $i \in \{1, 2, \ldots, k\}$. Devise a deterministic distributed algorithm for this problem with round complexity $O(D + k)$.

Solution: The algorithm idea is as follows. First, we find a BFS that spans the whole graph, e.g., by broadcasting the smallest id and every node choosing a node along the shortest path to the smallest-id node v as its parent. (The node with smallest id can be made aware of the fact that it has the smallest id by a suitable flooding-echo process (started at each node) where, during the flooding, nodes only forward the smallest id they have heard so far.) Notice that while broadcasting, the first neighbor forwarding the id of v is along the shortest path to v. After the BFS is constructed, each leaf node u can start sending values $x_1(u), \ldots, x_k(u)$ towards the root, one per round, ordered by index. For each index i, once a node u has received a value corresponding to this index from each child, it computes the minimum of the received values and its own value $x_i(u)$ and forwards it towards the root in the following round. (This minimum will be regarded as corresponding to index i.) Once the root has received all the minimum values, it computes the minimum of them (for each i), and broadcasts the values back to all nodes. Given that forwarding the values up and down the BFS tree does not suffer from congestion as each node sends the message corresponding to some index j precisely $j - i$ rounds after the message corresponding to index i, the time complexity is bounded by the time needed for constructing the BFS tree, and the sum of the depth of the tree and k, yielding $O(D + k)$.

(1b) Suppose there are k messages $m_1, m_2, \ldots, m_k$, each initially placed at an arbitrary node of the network (many or even all of the messages may be placed on the same node). Consider the following basic algorithm: per round, each node v picks one of the messages m_i that it has (from the beginning or received in the past) and sends m_i to all of its neighbors; node v will never send m_i again. Notice that a node will not send two of the messages at the same time. Prove that if we run this algorithm for $O(D + k)$ rounds, all nodes will receive all the messages.

Solution: We prove a slightly stronger statement, namely that if u is any node initially holding some message m and v any node in distance d from u, then v sends m at the latest in round $d + k$. This directly implies the desired statement.
Intuitively, what we want to argue is that m is sent by v not much later than round d (at the latest) or v sends many other messages up to that point. We prove this statement via induction in two variables, where the precise statement $S(d, \ell)$ for the induction is as follows:

For any integer $d \geq 0$, and any integer $0 \leq \ell \leq k$, after $d + \ell$ rounds, any node v in distance d from u has 1) sent m, or 2) sent at least ℓ messages.
The base cases are covered quickly: If $d = 0$, then if v does not send message m in the first ℓ rounds, then it must send ℓ other messages in those rounds since m is always available to be sent due to the fact that $v = u$; hence $S(0, \ell)$ holds for all $0 \leq \ell \leq k$. If $\ell = 0$, then $S(d, \ell)$ trivially holds.
For the induction step proving $S(d, \ell)$, we can assume $S(d-1, \ell)$ and $S(d, \ell-1)$ to hold. Consider node v after $d + \ell - 1$ steps. If v has sent message m or ℓ messages up to this point, we are done; hence assume that it has not, which implies that v has sent precisely $\ell - 1$ messages up to this point, by $S(d, \ell - 1)$. Now, consider a shortest path from u to v, and let v' denote the unique neighbor of v on this path. Due to $S(d-1, \ell)$, one of two cases must occur: v' has sent message m until (and including) round $d + \ell - 1$, or it has sent ℓ messages up to this point. Combining this insight with the fact that, before round $d + \ell$, node v has sent precisely $\ell - 1$ messages, neither of which is message m, we see that there is at least one unsent message available at v to be sent in round $d + \ell$. Hence, after $d + \ell$ rounds, v has sent ℓ messages, proving $S(d, \ell)$.

2 Minimum Spanning Tree

Consider an undirected connected graph $G = (V, E)$ where $n = |V|$. Suppose that each node $v \in V$ has selected one of its incident edges (v, u) as the *proposal edge of* v, let us denote it $e_v = (v, u)$. For instance, in the MST algorithm of Boruvka, this would be the minimum-weight edge incident on v. Notice that the two endpoints of an edge might propose this one edge simultaneously. Consider the random process that each node flips a fair coin for itself and then, we mark the proposed edge $e_v = (v, u)$ of node v only if v draws tail and u draws head.

Exercises

(2a) Prove that, in expectation, we mark at least $n/8$ edges, provided that $n \geq 2$.

Solution: Since every node selects a proposal edge and every proposal edge can be selected by at most two nodes, we get that at least $n/2$ distinct proposal edges are selected. Since the coin tosses are independent, we get that any proposal edge is marked with probability $1/4$. Let P be the set of proposal edges, so $|P| \geq n/2$, and let $X = \sum_{e \in P} X_e$, where X_e is an indicator random variable that has value 1 if e is marked and 0 otherwise. Given the above observations we can bound $\mathbb{E}[X] \geq |P| \cdot (1/4) \geq (n/2)/4 = n/8$.

(2b) Prove that, if we contract all the marked edges, the resulting graph has at most $7n/8$ nodes, in expectation, provided that $n \geq 2$.

Solution: For every contraction, at least one node is "removed" from the graph as a result of the contraction. Thus, the expected number of remaining nodes is at most $n - \mathbb{E}[X] \leq (7/8)n$.

(2c) Consider repeating the above process for $20 \log n$ iterations: In each iteration, we contract all the marked edges, and remove self-loops. Then, select one (e.g., min-weight) incident edge per new node, and repeat the marking process as above using one coin toss per each new node. Use (2b) to prove that, after $20 \log n$ iterations, with high probability, we have contracted everything to a single node.

Solution: Let X_i be the random variable whose value is the number of nodes after i iterations, minus 1. Since, for any number $k \geq 2$ of nodes at the beginning of an iteration, the number of nodes drops, in expectation, to a value that is at most $(7/8)k$ at the end of the iteration

(by (2b)), the expected value of "the number of nodes -1" at the end of the iteration is at most $(7/8)k - 1 < (7/8) \cdot (k-1)$. Also if we start with $k = 1$ nodes at the beginning of an iteration, "the number of nodes -1" drops by at least a factor of $7/8$ in expectation since it is 0 at the beginning and the end of the iteration. Hence, we have $\mathbb{E}[X_i] \leq (7/8) \cdot \mathbb{E}[X_{i-1}]$, for any $i \geq 1$. We obtain

$$\mathbb{E}[X_{20 \log n}] \leq n \cdot (7/8)^{20 \log n} \leq n \cdot (1/2)^{3 \log n} \leq n \cdot n^{-3} = 1/n^2 \ .$$

By Markov's inequality, it follows that $Pr(X_{20 \log n} \geq 1) \leq 1/n^2$. This implies, by the definition of our random variables, that the probability that more than one node remains after $20 \log n$ iterations is bounded from above by $1/n^2$.