

ASSIGNMENT 3 SOLUTIONS - DISTRIBUTED ALGORITHMS

- (1) We'll describe a β -like synchronizer; this synchronizer will use the existing trees $T \in \mathcal{T}$ instead of creating a spanning tree for the graph G . Since the synchronization problem can be reduced to the neighbours updating problem, we'll describe a solution to the latter.

We will perform a convergecast of the b_v bits of each $v \in V$ using all of the trees $T \in \mathcal{T}$, then broadcast all_v messages on them. Each vertex $v \in V$ will toggle its all_v bit only when it gets $Overlap_v(\mathcal{T})$ all_v messages (i.e. it gets a message from *all* of the trees it belongs to).

(Note that otherwise a vertex v might update its all_v bit before all of his neighbours w have updated their b_w bits!)

- $Time_{init} = Comm_{init} = 0$, since the trees are already given and no further preparations are needed.
- $Time_{pulse} = O(d \cdot l)$.
 d is the maximal diameter of a tree $T \in \mathcal{T}$ - and hence the maximal time needed for a pulse counter to change in the entire network. However, there is a congestion - each edge serves up to l trees - therefore, each message can be delayed for at most l time units.
- $Comm_{pulse} = O(l \cdot n)$: a vertex $v \in V$ sends $\Theta(Overlap_v(\mathcal{T}))$ messages on each pulse (Note that this number is not bounded by $|E|$ - an edge may be used more than once). Since there are n vertices with a maximal Overlap of l , the total amount of messages sent is $O(l \cdot n)$.
- $Time_{gap} \leq PulseDiff \cdot Time_{pulse} = 1 \cdot O(d) = O(d)$.

- (2) Use an α -like synchronizer, using the edges in H to pass messages. The problem here is that $(u, v) \in E$ does not imply that $(u, v) \in H$; say in other words, two neighbouring vertices in G may be up to k edges away in G' . The implication is that when a vertex v sends a message to its neighbours in G' , this message is received by vertices in G that are up to k edges away from v 's immediate neighbours.

Therefore, if we use the α synchronizer as-is, a vertex v might update its all_v bit before all of his neighbours w have updated their b_w bits.

To solve this, each vertex $v \in V$ will hold a counter c_v which is initialized to 0. When its b_v bit updates, it sends a message to its neighbours in G' ; when a vertex $u \in V$ gets such message it increments its counter c_u by one and, if $c_u < k$, sends this message to its neighbours in G' .

When $c_v = k$, the vertex v knows that each of its original neighbours in G have their b bits on.

- $Time_{init} = 0$
- $Comm_{init} = 0$

- $Time_{pulse} = O(k)$, since each vertex sends k messages to its neighbours in G' ; each of these messages requires $O(1)$ time steps.
- $Comm_{pulse} = O(h \cdot k)$, since each vertex v sends k messages to its $deg(v)$ neighbours in G' ($\sum_{v \in V} deg(v) = \Theta(|H|)$).
- $Time_{gap} \leq PulseDiff \cdot Time_{pulse} = O(Diam(G)) \cdot O(k)$. Note that two neighbouring vertices in G may be up to k edges away in G' , therefore their pulse difference between them can be $O(k)$.

- (3) (a) In the α synchronizer $|P_v - P_u| \leq 1$ for $(u, v) \in E$, hence $|P_v - P_u| \leq dist_G(u, v)$ for any $u, v \in V$.

In this case, $|P_{v'} - P_v| = |P_{v'} - 27| \leq dist(v_{11}, v_2) = 6$. Hence, $p_{v'} \in \{21, 22, \dots, 33\}$.

- (b) In the β synchronizer $|P_v - P_u| \leq 1$ for any $u, v \in V$, but note that a vertex v will increment its pulse number before the vertices in its rooted subtree.

Therefore, if v is the tree root, v' is a leaf and $p_v = 27$, there are two possibilities:

- $p_v = p_{v'} = 27$.
- $p_v = 27$, and v has issued a broadcast for the new pulse number, yet v' have not received it yet; hence $p_{v'} = 26$.

To summarize: $p_{v'} \in \{26, 27\}$.

- (c) In this case, when both v and v' are leaves, there are three possibilities:

- $p_v = p_{v'} = 27$.
- $p_{rt} = 27$, and rt has issued a broadcast for the new pulse number. v have got this message, so $p_v = 27$, yet v' have not received it yet, hence $p_{v'} = 26$.
- $p_{rt} = 28$, and rt has issued a broadcast for the new pulse number. v haven't received this message yet, so $p_v = 27$, but v' have received it, hence $p_{v'} = 28$.

To summarize: $p_{v'} \in \{26, 27, 28\}$.