In the following exercises we consider the CONGEST model. This is a synchronous message passing model with the additional property that the size of each message is bounded. If we assume that the nodes have IDs in \{1, \ldots, n \} and communicate by exchanging bitstrings, then each message is only allowed to contain \(O(\log n)\) bits. This means that each message may contain for example (the binary representation of) a constant number of integers \(\leq n^c\) for some constant \(c\). However, it is not possible that a node sends another node the IDs of all its neighbors in a single message, as the degree of the network may not be bounded.

Remark: Do not confuse the message size and the message complexity.

1. **\(k\)-Selection Problem in Graphs**

   Given a graph \(G\) with \(n\) nodes that have pairwise distinct input values \(\leq n^c\) for some constant \(c\), the \(k\)-selection problem for \(k \leq n\) is the problem of finding the \(k\)th-smallest value in the graph.

   Our goal is to describe a randomized distributed algorithm in the CONGEST model that solves the \(k\)-selection problem with an expected runtime of \(O(D \cdot \log n)\).

   a) Assume a tree \(T\) of depth \(D\). Describe an algorithm that computes in \(O(D)\) rounds for every node \(v\) a value \(s_v\) which equals the size (number of nodes) of the subtree with root \(v\).

   b) Assume a tree \(T\) of depth \(D\) and root \(r\) in which each node is able to flip coins. Describe a method to choose a node from the tree uniformly at random (i.e., each node has the same probability to be chosen) in time \(O(D)\).

      *Hint: Use the algorithm from a).*

   c) Describe a randomized algorithm that solves the \(k\)-selection problem with an expected runtime of \(O(D \cdot \log n)\).

      *Hint: Use the algorithm from b).*

2. **Leader Election**

   Given a graph \(G\), describe a deterministic algorithm in the CONGEST model such that every node learns the smallest ID in the graph and terminates after \(O(D)\) rounds. Analyse the message complexity of the algorithm.