University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
S. Faour, A. Mályusz

Theory of Distributed Systems
Exercise Sheet 3

Due: Wednesday, 15th of May 2024, 12:00 pm

In all the exercises of this problem set, we consider the synchronous message passing model on a graph, where nodes operate in synchronous rounds and all nodes start a computation together at time 0. We assume that initially, nodes do not know the IDs of their neighbors. Note that we will always denote by n the number of nodes in the graph and m the number of edges in the graph.

Exercise 1: Leader Election in General Graphs *(5 Points)*

Consider the following leader election algorithm. For simplicity, we assume that every node knows the graph diameter D. Every node u stores the largest ID it has seen in variable x_u. Each node $u \in V$ carries out the following algorithm.

Node u initially sets x_u :=ID(u) and sends x_u to its neighbors
for $D - 1$ rounds **do**
  if $x_v > x_u$ for the largest value x_v that u received **then**
    u sets $x_u := x_v$ and sends x_u to all neighbors from which it has not received a value equal to x_v
  end if
end for

After D rounds, the value x_u of each node u equals the largest ID in the network.

What is the message complexity of this algorithm (in terms of n)? Give an example that shows that your given bound is asymptotically tight (in the worst case) i.e. if $B(n)$ is your message complexity, give a family of graphs where your algorithm indeed has a message complexity of $\Omega(B(n))$.

Exercise 2: Leader Election via Radius Growth *(9 Points)*

We generalize the radius growth algorithm for leader election from the lecture to arbitrary graphs. Assume that every node knows the number of nodes in the graph. The algorithm consists of phases $i = 0, 1, 2, \ldots$. Let C_i be the set of leader candidates at the beginning of phase i. Set $C_0 = V$ (initially, each node is a leader candidate). Phase i of the algorithm consists of 2^i rounds. The algorithm terminates when $2^i \geq n$. In phase i, each node $u \in V$ carries out the following algorithm.

If $u \in C_i$, u initializes x_u :=ID(u) and sends x_u to its neighbors (otherwise, u initializes $x_u := -1$).
for $2^i - 1$ rounds **do**
  if $x_v > x_u$ for the largest value x_v that u received **then**
    u sets $x_u := x_v$ and sends x_u to all neighbors from which it has not received a value equal to x_v
  end if
end for
if $u \in C_i \wedge x_u ==$ID(u) **then**
  u joins C_{i+1} (i.e., u stays a candidate)
end if

---

a) Show that the number of messages sent in phase i is $O(\min\{2^i, |C_i|\} \cdot m)$.

b) Show that $|C_i| \leq \frac{4n}{2^i}$ for each phase i.

c) Show with a) and b) that the message complexity of the algorithm is at most $O(m\sqrt{n}\log n)$.

d) For $m = \Omega(n^2)$, the upper bound from c) becomes $O(n^{5/2}\log n)$. Give an example network on which the algorithm requires $\Omega(n^{5/2})$ messages.

Exercise 3: Leader Election in Complete Graphs *(6 Points)*

In a complete graph, one can trivially solve leader election in one round if every node sends its ID to all its neighbors. This requires $\Omega(n^2)$ messages. The following algorithm uses less messages at the cost of a slightly higher time complexity.

The algorithm consists of phases $i = 1, 2, \ldots$. Let C_i be the set of leader candidates at the beginning of phase i. Set $C_1 = V$ (initially, each node is a leader candidate). In phase i, each node $u \in V$ carries out the following algorithm.

---

if $u \in C_i$ **then**
  u sends a probe message (may I be a leader?) containing its ID to $\min\{2^i, n-1\}$ arbitrary neighbors.
end if
Let v be the node with the largest ID from which u received a probe message
if ID(v) >ID(u) **then**
  u sends back an acknowledgement to v
end if
if u received 2^i acknowledgements **then**
  u joins C_{i+1} (i.e., u remains a candidate)
end if

---

a) Argue that the algorithm solves leader election and analyze its time complexity.

b) Show that $|C_i| \leq \frac{n}{2^{i-1}}$ for each phase $i \geq 1$.

c) Analyze the message complexity.