Problem Set 10

This problem set is due **at 11:59pm** on **Friday, May 8, 2015.**

Each submitted solution should start with your name, the course number, the problem number, your recitation section, the date, and the names of any students with whom you collaborated.

Exercise 10-1. Read the lecture slides for Lectures L19 and L20.

Problem 10-1. Leader Election in a Synchronous Ring [25 points]

Consider a collection of n *identical* processes arranged in a synchronous ring network. Each process has two sets of ports, one to each of its immediate neighbors, with its ports to its clockwise neighbor named $left$ and its ports to its counterclockwise neighbor named $right$. Thus, the processes have a common sense of orientation.

The goal is for the processes to elect a single leader: exactly one process should eventually output LEADER.

(a) [5 points] First suppose that the processes are deterministic, and that they know n (the size of the ring). Either give a correct leader election algorithm for this case, or prove that no such algorithm exists. If you give an algorithm, analyze its time and message complexity.

(b) [10 points] Now suppose that the processes are probabilistic (i.e., randomized), and that they know n. We would like an algorithm that (a) never elects more than one leader, and (b) with probability 1, eventually elects a leader. Either give an algorithm satisfying these properties, or prove that none exists.

If you give an algorithm, analyze its time and message complexity. Your analysis should relate the complexity to the success probability. Specifically, for any ε, $0 < \varepsilon < 1$, you should provide bounds on the time and message complexity that hold with probability at least $1 - \varepsilon$.

(c) [10 points] Finally suppose that the processes are probabilistic and they do not know n. That is, the same algorithm must work regardless of the size of the ring in which the processes are placed. We would again like an algorithm that (a) never elects more than one leader, and (b) with probability 1, eventually elects a leader. Either give an algorithm satisfying these properties, or prove that none exists. If you give an algorithm, analyze its time and message complexity as described in Part (b).

Problem 10-2. Breadth-First Search in an Asynchronous Network [25 points]

The asynchronous Breadth-First Search algorithm presented in class involves corrections that could trigger the sending of many messages, resulting in worst-case message complexity $O(nE)$ and worst-case time complexity $O(diam \cdot n \cdot d)$, until all nodes' $parent$ variables have stabilized to correct parents in a breadth-first spanning tree. (We are not considering individual processes' $parent$ outputs here, nor global termination. We are also ignoring local processing time.)

This problem involves designing a better asynchronous Breadth-First Search algorithm, one that does not make any corrections. Thus, once a process sets its $parent$ variable, it can output the value since that is its final decision.

Assume that the network graph is connected and contains at least two nodes.

(a) [18 points] Describe carefully, in words, an algorithm in which the root node v_0 coordinates the construction of the tree, level by level. Your algorithm should take time $O(diam^2 \cdot d)$ until all the $parent$ variables are set.

(*Hint:* The root node can conduct broadcast-convergecast waves to build the successive levels in the tree. Four types of messages should be enough, e.g., $search$ messages that test for new nodes, $parent(b)$ (b a Boolean) for parent/nonparent responses, and $ready$ and $done$ messages for broadcasting and convergecasting signals on the tree.)

(b) [7 points] Analyze the time and communication complexity of your algorithm, and compare them to the costs of the asynchronous BFS algorithm presented in class.

6.046J / 18.410J Design and Analysis of Algorithms
Spring 2015