Problem Set 7

Due: Thursday, December 5, 2013

Readings:
Dolev book, Chapter 2

Next week:
Chapters 23-25; Fan’s thesis, clock synchronization lower bound; various papers on distributed algorithms for dynamic networks.

Problems:

1. Consider the problem of establishing and maintaining a shortest-paths tree in a network with a distinguished root node $i_0$ and with weights associated with the edges. The problem is similar to the one studied in Section 15.4 of the textbook, except that now we model the channels as single-writer, single-reader registers, as Dolev does for his basic BFS spanning tree algorithm (see his Section 2.5). Here we consider self-stabilizing algorithms to solve the shortest-paths problem.

(a) Assume that the weights on the edges are fixed, and are known by the processes at the endpoints. Moreover, these weights do not get corrupted. Write pseudocode for a self-stabilizing algorithm that establishing a shortest-paths tree. You may use the style from the Lynch textbook, or the Dolev book.

(b) Give a proof sketch that your algorithm works correctly; i.e., that it in fact stabilizes to a shortest-paths tree.

(c) State and prove an upper bound on the stabilization time.

(d) Describe how your algorithm (or a simple variation) can be used in a setting in which the costs on the edges change from time to time. State a theorem about the behavior of your algorithm in this setting. State your assumptions clearly.

2. Consider the 2-Exclusion Problem, which allows two processes to coexist in the critical region, and requires that, when one process stops in the critical region, a second process should still be able to advance to the region. (Recall that we studied the $k$-exclusion problem when we studied ordinary mutual exclusion—see, for example, Exercise 10.13.)

(a) Define carefully the “Self-Stabilizing 2-exclusion Problem”, for a model like the one Dijkstra used for his Self-Stabilizing Mutual Exclusion algorithm.

(We covered Dijkstra’s model and algorithm in class, and it’s in Dolev’s book, Section 2.6.)

(b) Describe a self-stabilizing algorithm that solves your 2-exclusion problem, and prove that it works.