Problem Set 7

Due: Thursday, December 10, 2009

Reading:

Borowsky, Gafni, Lynch, Rajsbaum paper.
Attiya, Welch, Section 5.3.2 (optional).
Attie, Guerraoui, Kouznetsov, Lynch, Rajsbaum paper on boosting fault-tolerance.
Chapter 17 of Lynch's book.
Lamport's "Part-Time Parliament" paper.

Reading for next week:

Dolev's book on self-stabilization, especially Chapter 2
Chapters 23-25 of Lynch's book
Attiya, Welch, Section 6.3 and Chapter 13

Problems:

1. As noted in class, the BGLR paper has a liveness bug in the main protocol. Namely, a simulating process i may repeatedly decide to select the same process j to perform a snapshot, using safe-agreement, neglecting some other process j'.

   (a) Why doesn't the task structure of process i, which has a separate task for each simulated process, ensure progress for all the simulated processes?

   (b) Give a simple modification to the given code that would fix this problem, and guarantee that all the simulated processes get fair turns.

2. Consider a version of the *approximate agreement* problem, expressed as a decision problem as follows. The value domain V is the set of rational numbers. For any input vector I of elements of V, the allowable output vectors are those for which (a) every output is in the range of the input values in I, and (b) the difference between any two output values is at most $\frac{\Delta}{99}$, where Δ is the maximum difference between any two values in I.

   (a) Consider an asynchronous read/write shared-memory system with 100 processes and at most 1 stopping failure. Describe a (very, very simple) algorithm A that solves the given approximate agreement problem for this model.

   (b) Now apply the BG transformation to obtain an algorithm for 2 processes and at most 1 stopping failure. Does the resulting algorithm solve the given approximate agreement problem? Explain why or why not.

3. For each of the following pairs of resilient (fault-tolerant) atomic objects, A and B, say whether or not A can be implemented using an unlimited number of B's, plus an unlimited number of reliable read/write registers. Prove your answers.

   (a) A is an 8-process 3-resilient consensus atomic object, and B is a 4-process 2-resilient consensus atomic object.

(b) A is an 8-process 4-resilient 2-consensus (AKA 2-set-consensus) atomic object, and B is a 4-process 3-resilient consensus atomic object. (For this part and the next, it will prove useful to consult Section 5 of the AGKLR paper.)

(c) A is an 8-process 4-resilient 4-consensus atomic object, and B is a 4-process 2-resilient consensus atomic object. (Hint: you may find it helpful to connect processes to more than one object.)

4. In the first phase of the Paxos consensus algorithm, a participating process i performs a step whereby it abstains from an entire group of ballots at once; namely, the set B of all ballots whose identifiers are less than some particular proposed ballot identifier b, and that i has not already voted for. This set B may include ballots that have not yet been created.
Suppose that, instead, process i simply abstained from all ballots in the set B that it knows have already been created. Does the algorithm still guarantee the agreement property? If so, give a convincing argument. If not, give a counterexample execution.

5. Consider the problem of establishing and maintaining a shortest-paths tree in a network with a distinguished root node i_0, and costs associated with the edges. The problem is similar to the one studied in Section 15.4 of Lynch's book, except that, now, we model the channels as registers; i.e., as Dolev does for his basic BFS spanning tree algorithm (see his Section 2.5). In this problem, we consider self-stabilizing algorithms to solve the shortest-paths problem.

   (a) Assume that the costs on the edges are fixed, and known by the processes at the endpoints, and that these costs do not get corrupted. Write code, either in Tempo-style (precondition-effect code) or in Dolev's style, for a self-stabilizing algorithm that maintains a shortest-paths tree.

   (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. Be sure to state your assumptions clearly.

6. Describe, in no more than half a page, an interesting research project you could imagine that you (or another student) could do, related to something (anything) we have studied in 6.852 this term. Be creative.

6.852J / 18.437J Distributed Algorithms

Fall 2009