Concurrent Algorithms: Exercise 2

October 5, 2009

Problem 1

1. Devise an algorithm that implements an atomic M-valued SWMR register using (any number of) atomic binary SWMR registers.
2. Prove that your algorithm is correct.
3. Explain whether your algorithm remains correct (i.e., implements an atomic register) if you change the binary base registers from atomic to regular.

Problem 2

Consider the *binary consensus* problem from the previous exercise sheet. Recall that this abstraction satisfies the following properties:

Agreement No two processes decide different values.

Validity The value decided is one of the values proposed.

Let's assume that processes might in fact *crash*, i.e. stop taking steps, at any point during the execution of an algorithm. We add an extra condition on progress:

Termination Every process that does not crash will eventually decide.

Assume the following theorem is true:

Theorem 1 (FLP) *Binary consensus is impossible among N processes, if one of them might fail by crashing, in an asynchronous system that disposes of binary SRSW safe registers*[1].

Using what you know about registers, prove the following result:

Theorem 2 (Students' FLP) *Binary consensus is impossible among N processes, if one of them might fail by crashing, in an asynchronous system that disposes of MRMW atomic registers.*

[1] In fact, this is one of the major results in distributed computing, first stated by Fisher, Lynch and Patterson, in their paper "Impossibility of Distributed Consensus with One Faulty Process" in 1985. For details, see the presentation in the Encyclopedia of Algorithms (available on Google Books), or the original paper, which is quite readable.