Problem Set 2, Part b

Due: Thursday, October 6, 2011
Problem sets will be collected in class. Please hand in each problem on a separate page.

Readings:

Chapter 7 (skim 7.2)
Chapter 8

For next week: Chapters 14, 15.

Problems:


7. This exercise is designed to illustrate the Sperner construction, but for a really trivial case: \( n = 3, \ k = 2, \ r = 0, \ f = 0 \). That is, we are considering 2-agreement among 3 processes, none of which are faulty, and where the algorithm proceeds for 0 rounds. This means that the processes don’t even exchange any messages, but must decide based just on their initial values. Specifically, suppose each process decides on its own initial value.

   This is obviously an incorrect algorithm, and in fact, it should be pretty obvious that no algorithm could work for this case. But let’s see how the Sperner construction demonstrates its incorrectness.

   The Bermuda triangle for this case has the following structure:

   It can be labelled with 0-round “executions”, each of which is just a vector of initial values, as follows:

   (a) Label each tiny triangle in the diagram just above with a 0-round execution (vector of initial values), and label each node \( v \) with a process index \( i_v \in \{1, 2, 3\} \), in such a way that the execution inside each triangle is indistinguishable from the execution at each corner vertex \( v \) of the triangle by process \( i_v \).

   (b) Label each vertex \( v \) with the decision that process \( i_v \) makes in the execution that labels vertex \( v \). For this, you should use the specific trivial algorithm described above, in which everyone decides on its own initial value.

   (c) Indicate a tiny triangle in which the decisions labeling the three corners are different.

   (d) Explain briefly why this demonstrates that the given algorithm is incorrect.

8. Exercise 8.5. Write your algorithm using the pseudocode style used in the book. If you have time, we encourage you to write the (non-task portions of the) code using Tempo. This is not required for this assignment, but will be required for Problem Set 3.

9. This exercise describes two ways in which one can modify an I/O automaton. In each case, the modification restricts the form of the automaton without introducing any new fair traces.

   (a) Exercise 8.8.

   (b) Exercise 8.9.