ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

HS 2011 Prof. R. Wattenhofer / B. Keller

Distributed Systems Part II

Exercise Sheet 1

1 An asynchronous Riddle

A hangman summons his 100 prisoners, announcing that they may meet to plan a strategy, but will then be put in isolated cells, with no communication. He explains that he has set up a switch room that contains a single switch. Also, the switch is not connected to anything, but a prisoner entering the room may see whether the switch is on or off (because the switch is up or down). Every once in a while the hangman will let one arbitrary prisoner into the switch room. The prisoner may throw the switch (on to off, or vice versa), or leave the switch unchanged. Nobody but the prisoners will ever enter the switch room. The hangman promises to let any prisoner enter the room from time to time, arbitrarily often. That is, eventually, each prisoner has been in the room at least once, twice, a thousand times or any number you want. At any time, any prisoner may declare "We have all visited the switch room at least once". If the claim is correct, all prisoners will be released. If the claim is wrong, the hangman will execute his job (on all the prisoners). Which Strategy would you choose...

a) ...if the hangman tells them, that the switch is off at the beginning?

b) ...if they don't know anything about the initial state of the switch?

2 Communication Models

Shared memory allows multiple processes to read and write data from the same location. **Message passing** is another way for processes to communicate: each process can send messages to other processes. Compare these models: where do they differ and where are they similar? You might consider different models of message passing, for example with or without message loss.

3 Communication without Computers

Consider the actions described below, in which model (shared memory or message passing) can you describe them best? Why and how?

- Communication via postcards
- Two people speaking in a room
- Instant messages via Skype (data remains on client if partner is offline)
- Many people speaking in a room

4 Consensus with an n-Register

A *n-register* allows up to n registers to be written at the same time. Processes may still only read one value at a time. Let $n = 6$, give an asynchronous and wait-free protocol which solves consensus for 3 processes. You may assume the registers are initialized with -1 and processes have a unique id.

5 Consensus for two Processes

A lousy programmer wanted to solve consensus for 2 processes and came up with a sophisticated protocol. Does the protocol really solve consensus? Simplify the protocol and then proof or reason your claim. If the protocol does not reach consensus provide a counterexample. If it does, draw the important part of the execution tree.

```
initialize(){
  // s  and i are shared
 s = '?';
 i = 0;

  // the input, an array of length 2
  input[] = [random({0,1}), random({0,1})];
}

// making the decision
decide(){
  // the id of this process, 0 or 1
  id = this.getThreadId();

  decisionMade = false;
  decision = input[id];

  while(decisionMade == false){
    value = s; // read s
    if(value == '?'){
      s = input[id]; // write s
    }
    else if(value != decision){
      decisionMade = true;
      decision = value;
    }
    else{
      if(i.fetchAndInc() == 1){
        decision = input[ 1-id ];
      }
      decisionMade = true;
    }
  }
}
```