Tutorial sheet: Model-Checking

The following questions illustrate typical ones that you may get in the exam. They are not meant to be finished in 30 minutes, and some of the later questions may be rather difficult. However, you should find it to be good practice for the exam. If you can solve this kind of question, you are well prepared for the model-checking exam question.

1. Consider the following concurrent program:

   ```
   vars x, y: Boolean;

   while (not y) { x := not y; } |||
   while (not x) { y := not x; }
   ```

   where ||| interleaves the two programs arbitrarily.

   Without constructing the automaton, explain how this program can be converted into a finite state automaton. Explain carefully what information the states would contain, and calculate how many states the resulting automaton would have (global number of states, not reachable). Assuming that Boolean variables start off false, show the initial state of the system.

2. Express in terms of the information stored in the state, the following properties:

   a. If \( x \) is true, then so is \( y \).
   b. \( x \) and \( y \) are mutually exclusive (they are never both true at the same time).
   c. If \( x \) and \( y \) are both true, then the program has terminated.
   d. If \( x \) is false, then the second program has not yet terminated.
   e. When at least one of the programs has just entered the loop, \( x \) and \( y \) are mutually exclusive.

3. Give a counter-example trace to show that the property (a) is not true.

4. Do you think that property (b) would be verified by induction with depth 1? Justify your answer by either giving a transition which would not allow depth 1 induction to verify the system, or by arguing why induction would work.

5. Property (a) cannot be verified using induction with depth 1. Show why and explain what you would conclude about the system from this statement.

6. ‘Abstract interpretation is applied to the automaton by collapsing all states in the same program location, and with equal values of \( x \) into the same abstract state’. Give two states that were different and have now been collapsed into one.

7. Construct a counter-example for property (b) which is a false-negative.
Model answers:

1. We will need two program counters and the values of the two variables. We can number the program locations as follows:

   0: while (not y) {
   1:   x := not y; }
   2:

   0: while (not x) {
   1:   y := not x; }
   2:

   The programs starts off at location 0. After making the while condition test they either go to location 1, if it is satisfied, or location 2 if it is not. At location 1 they execute the assignment and go back to location 0. Location 2 is the termination location.

   The resultant transition system states will thus be 4-tuples:
   (left program counter, right program counter, value of x, value of y)
   and will have 3*3*2*2=36 states.

   The only initial state is (0,0,false,false).

   Note: If you added other locations in the program (eg one inside the loop body after the assignment), the answer would still be correct.

2. Given a state (pc_L, pc_R, x, y), the properties are:

   a. x => y
   b. not (x and y)
   c. (x and y) => (pc_L=2 and pc_R=2)
   d. not x => pc_R%2
   e. (pc_L=1 or pc_R=1) => not (x and y)

3. The following is a 5 step counter example:

   (0,0,false,false)  
   → (0,1,false,false)  
   → (1,1,false,false)  
   → (0,1, true,false)

4. Yes, property (b) is satisfied, and would be proved correct using depth 1 induction. To verify the property using depth 1 induction we would have to verify two things:
a. The initial states satisfy the property: Since the only initial state is 
(0, 0, false, false), this is clearly satisfied.

b. Performing a transition from a good state cannot take us to a bad state: 
Intuitively, every step in the program either leaves the variables 
unchanged, or an assignment is made. Since the property depends only on 
the variables, in the first case it would still be satisfied. In the second case, 
either $x$ takes on the opposite value of $y$, or vice-versa. This can never 
lead to both variables being true. (If you cannot see why, try all cases of $x$
and $y$).

5. Consider the transition from (1,1,false,false) to (2,1,true,false). 
The source state is correct, but not the destination. 

Nothing can be concluded. Since the problem is not with the initial state, we 
would have to try deeper induction.

6. Two states $pc_{1L}$, $pc_{1R}$, $x_{1}$, $y_{1}$ and $pc_{2L}$, $pc_{2R}$, $x_{2}$, $y_{2}$ are now 
considered equivalent, if:

$$pc_{1L}=pc_{2L} \text{ and } pc_{1R}=pc_{2R} \text{ and } x_{1}=x_{2}$$

An example of a pair of states which were not equivalent, but have now been 
collapsed into the same one are: (0,0,false,false) and 
(0,0,false,true).

7. The following is a trace in the abstracted automaton:

$$(0,0,\text{false, false}), (0,0,\text{false, true})$$
$$\rightarrow (0,1,\text{false, false}), (0,1,\text{false, true})$$
$$\rightarrow (1,1,\text{false, false}), (1,1,\text{false, true})$$
$$\rightarrow (0,1,\text{true, false}), (0,1,\text{true, true})$$

Note that (a) the transitions are allowed (there is always a transition between the 
first elements of the sets as written) (b) the last state is a bad state because its 
second element is a bad state in the original system (c) the first set is an initial 
state in the abstract system (since its first element is an initial state in the original 
system) (d) it is a false-negative, since there is no concrete trace in the original 
automaton corresponding to the abstract trace which leads to a bad state.

If the answers are not clear, or you need further explanations, send me an e-mail 
at gpace@mcst.org.mt or call me on 2360 2139 Good luck! 

Gordon