Exam Software Verification (TDDE34)

Jour: Ahmed Rezine (1938).

- You may answer in either English or Swedish.
- Write clearly. **Unreadable text will be ignored.**
- Be precise in your statements. Ambiguous formulations will lead to reduction of points.
- Motivate clearly all statements and reasoning.
- The exam is 40 points and graded U, 3, 4, 5 (**preliminary** limits: 20p, 30p, 35p).
  Points are given for motivations, explanations, and reasoning.

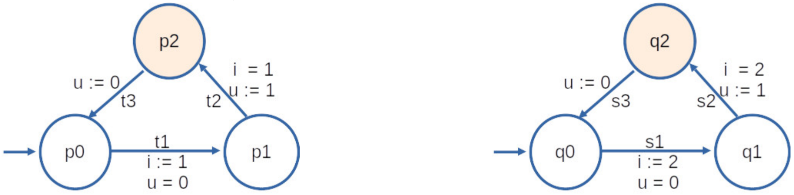
1 Branching time (10p)

Assume `Req`, `Ack` and `Reset` are atomic propositions. We use ! for negation. Recall **EX**(P) means P holds on at least one state directly following the current state. Express the following CTL properties using (boolean combinations of) true, **E**, **U**, **X**, `Req`, `Ack` and `Reset`:

- **EF**(`!Ack`) (2p)
- **AX**(`Ack`) (2p)
- **AG**(**EF**(`Reset`)) (3p)
- **AG**(`Req`) $\Rightarrow$ **AG**(`Ack`) (3p)

2 Mutual exclusion (12p)

Assume the following description of a simple mutual exclusion algorithm for two processes **p** and **q**. State **p0** (resp. **q0**) is the initial state of process **p** (resp. process **q**). Process **p** (resp. process **q**) moves to state **p1** (resp. state **q1**) in case it wants to access its critical section. State **p2** (resp. **q2**) is the critical section of process **p** (resp. process **q**). Variables **i** and **u** are shared by the two processes. Initially, **i** is 1 and **u** is 0. Variable **i** is set to 1 by process **p** in transition **t1**, and to 2 by process **q** in transition **s1**. Variable **u** is set to 1 by processes **p** and **q** when they enter their respective critical sections (transitions **t2** and **s2**). It is reset to 0 on exit (transitions **t3** and **s3**). Each transition is atomic. For instance **t1** both tests if **u** is 0 (with condition **u=0**) and sets **i** to 1 (with assignment **i:=1**).



In the following, we use **@pi** to mean the proposition stating process **p** is at state **pi**. We do the same for process **q**. For instance, proposition **@p1** is true in a configuration when process **p** wants to access the critical section and proposition **@q2** is true when process **q** is at its critical section. We use the following set of atomic propositions:

- Location propositions: $\{@\mathbf{pi} \mid 0 \leq \mathbf{i} \leq 2\} \cup \{@\mathbf{qi} \mid 0 \leq \mathbf{i} \leq 2\}$
- Values' propositions: $\{\mathbf{i = 1}, \mathbf{i = 2}, \mathbf{u = 0}, \mathbf{u = 1}\}$

Answer the following questions:

- Give an LTL formula that states mutual exclusion for the processes **p** and **q**. Recall mutual exclusion states that the two processes can never be both at their critical sections at the same time. (2p)

- Do processes **p** and **q** above respect mutual exclusion? if they do, argue why, otherwise give an execution that violates it (3p)
- Write an LTL formula φ corresponding to the starvation freedom of **p**, i.e., each time **p** wants to access its critical section it eventually succeeds. Your formula should NOT use value propositions $\{\mathbf{i = 1}, \mathbf{i = 2}, \mathbf{u = 0}, \mathbf{u = 1}\}$. (2p)
- Give a Büchi automaton for the formula φ. Explain it. (3p)
- A good scheduler is a scheduler that selects each one of the two processes infinitely often. In other words, a good scheduler cannot start ignoring a process forever. Would having a good scheduler guarantee starvation freedom of **p**? Explain or give an execution that violates it. (2p)

3 Symbolic representation (8p)

1. Assume integer variables x_1, x_2 and x_3. A pure function f that associates integers to integers (assume the domain of f contains all integers). An integer-indexed array *Arr* containing integer values (assume the size of *Arr* is infinite and all integers are valid indices). The following formula involves expressions in the Linear Integer Arithmetic (LIA), the Equality over Uninterpreted Functions (EUF) and the Arrays (A) fragments. Recall $wr(Arr, x_2, x_3)$, for an array *Arr* and two integers x_2 and x_3, is an array that coincides with *Arr* on all indices except possibly for the cell with index x_2 that is assigned with value x_3. Also, $rd(Arr, 3)$ is the value of the cell with index 3 in the array *Arr*.

$$\left(\begin{array}{c} (x_1 \leq 2 \wedge 3 \leq x_2 \wedge x_1 + 1 = x_2) \\ \wedge \\ ((f(x_1) = f(2)) \Rightarrow (rd(wr(Arr, x_2, x_3), 3) = (x_1 + x_3))) \end{array}\right)$$

   Give a model (i.e., values for the variables and the relevant array cells and function mappings) for the formula if it is satisfiable, otherwise argue why it is not satisfiable. (4p)

2. Consider the formula $f(v_0, v_1, v_2, v_3)$ defined as $((v_0 \oplus v_1) = (v_2 \vee v_3))$ where v_0, v_1, v_2 and v_3 are boolean variables, $\oplus$ stands for the "exclusive or" binary operator and $\vee$ stands for the "or" binary operator. Give a BDD for f assuming the order $v_0 < v_1 < v_2 < v_3$ (i.e., starting from the root, variable v_0 appears first, then variable v_1, ... etc). (4p).

4 Partial and total correctness (10p)

Consider the following simple program (all variables are natural numbers):

```
$\{Q : r = 1 \land p = 1 \land n \geq 1 \land q \geq 2\}$
 $\textbf{while}(p < n)\{$
    $r := r + q^p;$
    $p := p + 1;$
 $\}$
$\{R : r = (q^n - 1)/(q - 1)\}$
```

- Find a suitable invariant and use it to show that if the loop terminates after starting from a state satisfying Q then it terminates in a state satisfying R (6p)
- Find a suitable variant function and use it to show the loop terminates. (4p)