Exam Software Verification

June 3, 2022

- Time kl 08.00 - 12.00
- Submit your answers by email to `ahmed.rezine@liu.se`
- This is an open book exam. You can access internet.
- It is however strictly forbidden to contact and discuss the exam, during the exam perion, with any person other than the examiner, whether the person is related to the course or not.

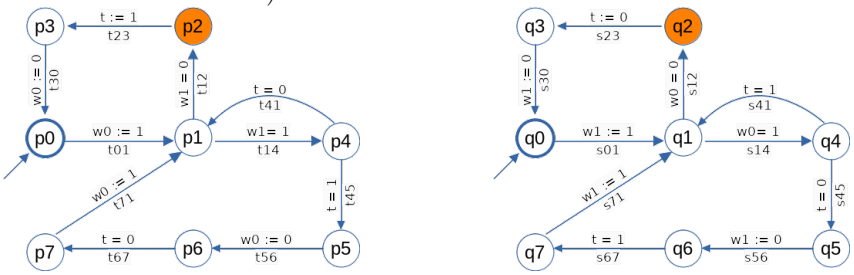
1 Branching time (4p)

Assume `Fail`, `Req`, `Ack`, `DeviceEnabled` and `Restart` are atomic propositions. Express the following CTL properties using (boolean combinations of) **EG**, **EU** and the atomic propositions:

- **EF**(`Fail`) (1p)
- **AG**(`Req` $\Rightarrow$ **AF**(`Ack`)) (1p)
- **AG**(**AF**(`DeviceEnabled`)) (1p)
- **AG**(**EF**(`Restart`)) (1p)

2 Mutual exclusion (8p)

Assume the following description of Dekker's mutual exclusion algorithm for two processes **p** and **q**. State **p0** (resp. **q0**) is the initial state of process **p** (resp. process **q**). State **p2** (resp. **q2**) is the critical section of process **p** (resp. process **q**). Variable **w0** is only written by process **p**. It is 1 when **p** wants to access its critical section (**p2**). Similarly, variable **w1** is only written by process **q**. It is 1 when **q** wants to access its critical section (**q2**). Variables **w0, w1, t** take their values in **{0,1}**. Variable **t** is read and written by both processes. Transitions are either tests (e.g. **w1=0** for transition **t12**) or assignments (e.g. **w1 := 0** for transition **s56**).



2.1 Part A

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, the proposition **@q2** is true in a configuration when process **q** is at its critical section. We use the following set of atomic propositions:

- Location propositions: $\{@\mathbf{pi} \mid 0 \leq \mathbf{i} \leq 7\} \cup \{@\mathbf{qi} \mid 0 \leq \mathbf{i} \leq 7\}$
- Values' propositions: $\{\mathbf{x} = \mathbf{v} \mid \mathbf{x} \text{ in } \{\mathbf{w0}, \mathbf{w1}, \mathbf{t}\} \text{ and } \mathbf{v} \text{ in } \{\mathbf{0}, \mathbf{1}\}\}$

Answer the following questions:

- Write an LTL formula φ_{mx} that states that mutual exclusion is always respected. (2p)

- Write an LTL formula φ_{eat} that states that each time process **p** wants to access its critical section it eventually succeeds. (2p)
- Give a Büchi automaton for the formula φ_{eat}. Explain it. (2p)

2.2 Part B

We assume the transitions are atomic. Transitions from different processes can be interleaved (a scheduler schedules one process at a time to execute a number of transitions). Transitions corresponding to assignements (e.g., **t01** or **s71**) are enabled if the corresponding process is at the start of the transition (e.g., **@q7** holds for **s71**). Transitions corresponding to tests (e.g., **t14** or **s45**) are enabled if the corrsponding process is at the start of the transition and the test is true (e.g., **@q4** and **t=0** for **s45**). We write **En(t)** to mean transition **t** is enabled. We write **Ex(t)** to mean transition **t** is indeed executed. For instance **Ex(s45)** is true if **En(s45)** and process **q** moves from **q4** to **q5**. To simplify the discussion, we will hereafter discuss LTL formulas over $\{\mathbf{En}(\mathbf{t}) \mid \mathbf{t} \text{ is a transition}\}$ and $\{\mathbf{Ex}(\mathbf{t}) \mid \mathbf{t} \text{ is a transition}\}$. You should not use the atomic propositions from part A. It is reasonable to assume schedulers behave "reasonably". A way to account for this assumption is to restrict runs to those satisfying a "reasonable" constraints. Consider the following constraint:

Φ: for all transition **u** of processes **p** and **q**. GF(!**En(u)** or **Ex(u)**)

- Is restricting scheduler's behavior to Φ enough to ensure φ_{eat}? explain. (2p)

3 Symbolic representation (6p)

Consider the formula $f(v_0, v_0', v_1, v_1', v_2, v_2') = (v_0' = \neg v_0) \wedge (v_1' = v_0 \oplus v_1) \wedge (v_2' = (v_0 \wedge v_1) \oplus v_2)$ where $v_0, v_0', v_1, v_1', v_2$ and v_2' are boolean variables and $\oplus$ is exclusive or. Give a BDD for f assuming the order $v_0 < v_0' < v_1 < v_1' < v_2 < v_2'$ (i.e., starting from the root, variable v_0 appears first, then variable v_1, ... etc).

4 Partial and total correctness (6p)

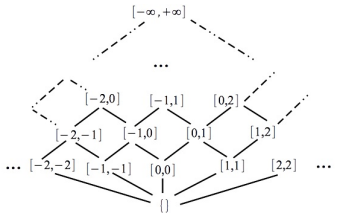
Consider the following simple program:

$$
\begin{array}{l}
\{Q : x = 0 \wedge y = 0\} \\
\ \textbf{do} \quad x < 100 \quad \rightarrow \quad x := x + 1; y := y + 2 \\
\ \textbf{od} \\
\{R : y < 201\}
\end{array}
$$

- 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 (4p)

- Find a suitable variant function and use it to show the loop terminates. (2p)

5 Abstract Interpretation (6p)



We adopt the following widening operator ∇ for the interval domain:

- $[a, b]\nabla\bot = \bot\nabla[a, b] = [a, b]$
- $[a, b]\nabla[c, d] = [l, r]$ with
  - $l = a$ if $a \leq c$ and $l = -\infty$ otherwise
  - $r = b$ if $b \geq d$ and $r = +\infty$ otherwise

- Give a sequence of intervals obtained durinf a fixpoint computation where you systematically use widening as a join operator. (4p)
- The obtained fixpoint does not establish that `x <= 101` at line `L4`. Describe how such a fact can be established using the interval domain. (2p)

```
//x: ⊤ = [−oo, +oo]
L1. x:= 0
//x: ⊥
L2. x:= x + 1
//x: ⊥
L3. if x < 100 goto L2
//x: ⊥
L4. nop
//x: ⊥
L5. end
```