UNIVERSITY OF BIRMINGHAM

School of Computer Science

First Year – Undergraduate Affiliate Computer Science/Software Engineering
Third Year – BSc Computer Science
Third Year – MSci Computer Science
Third & Fourth Year – MEng Computer Science/Software Engineering
Third Year – BSc Computer Science with Industrial Year
Third Year – MEng Computer Science/Software Engineering with Industrial Year

06 28201

Computer-Aided Verification

Summer May/June Examinations 2016

Time allowed: 1 hour 30 minutes

[Answer ALL Questions]

1. Below is a simple concurrent program, comprising two independent parallel processes accessing a shared integer variable x, which is initially set to 0.

```
l_0 :  while (x ≠ 1) {
l_1 :      x := 2 − x;
       }
l_2 :  end
```

```
l_0 :  x := 1;
l_1 :  end
```

For convenience, lines of the program are labelled with program locations (of the form l_i).

(a) Draw a labelled transition system representing the system described above, where, apart from the mutual dependency on the variable x, the two processes operate asynchronously.

[10%]

(b) For each of the following properties, explain how it can be expressed in the specified temporal logic, with respect to the labelled transition system (LTS) above, state whether it is satisfied in the LTS and, if it is not, give a counterexample.

(i) x is always greater than or equal to 0 (in LTL);

(ii) x eventually becomes equal to 1 (in LTL);

(iii) it is always possible to reach a state where $x > 0$ (in CTL);

(iv) x takes the value 2 infinitely often (in LTL);

(v) in any execution where x eventually equals 1, x is equal to 2 only finitely often (in LTL).

[20%]

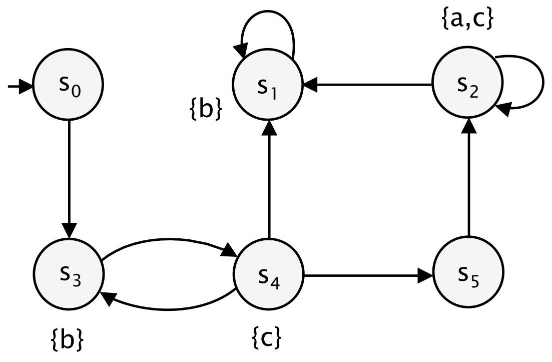
2. (a) Describe two differences between the logics CTL and LTL. **[4%]**

   (b) For which of the two logics LTL and CTL (if any) are these legal formulae? (You can assume that a, b and c are atomic propositions)

   i. $(\exists\Box\neg b) \wedge \forall\bigcirc\neg(a \wedge c)$
   ii. $\Diamond\Box\Diamond(a \wedge b)$
   iii. $b \wedge a\Diamond(b \rightarrow \bigcirc c)$
   iv. $(\Box a) \rightarrow \bigcirc(b \,\mathtt{U}\, c)$
   v. $\forall(a \,\mathtt{U}\, \exists b)$
   vi. $\exists\bigcirc\, \forall\Box\neg a$

   [6%]

   (c) Choose *one* of the legal CTL formulae given in part (b) above, and illustrate the application of the CTL model checking algorithm to compute the set of states of the labelled transition system below which satisfy it.



[10%]

3. Here are two properties expressed in the temporal logic LTL:

   (i) $\Box(a \rightarrow \bigcirc b)$

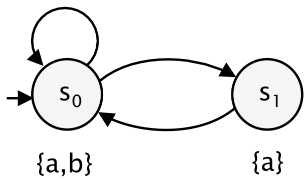
   (ii) $\Box(a \rightarrow \Diamond b)$

   where a and b are atomic propositions.

   (a) For each of the two formulae above, state whether it represents a safety property and justify your answer.

   [7%]

   (b) Illustrate the model checking procedure for verifying whether property (i) above is satisfied by the labelled transition system (LTS) shown below, by constructing an appropriate LTS-automaton product.



   [10%]

   (c) Property (i) above requires b to be true immediately after a; whereas property (ii) requires b to be true at some point in the future. An alternative property might be $\Box(a \rightarrow \Diamond^{\leq k} b)$, which requires b to be true within k steps.

   Formally, this uses a new temporal operator $\Diamond^{\leq k}$, which can be added to the existing semantics for LTL as follows. For integer $k \geq 0$, LTL formula ψ and any infinite trace σ, we have:

$$\sigma \models \Diamond^{\leq k} \psi \quad \Leftrightarrow \quad \exists j \leq k \text{ such that } \sigma[j \ldots] \models \psi$$

   Does this new operator $\Diamond^{\leq k}$ add expressive power to LTL? Justify your answer.

   [8%]

4. (a) Explain two advantages and two disadvantages of model checking, with respect to alternative methods for formal verification.

[5%]

(b) Illustrate the main steps of applying *bounded model checking* to verify the correctness of the following program fragment which manipulates two integer variables, i and j, and accesses a third integer variable n. The values of j and n are unknown. Here, "correctness" means that the assertion is never violated.

```
for ($i := 1$; $i \leq n$; $i{+}{+}$) {
    $j := j + i$;
}
assert $j < 10$;
```

You should include:

- simplification of control flow;
- loop unwinding (do this to a depth of 2);
- conversion to single static assignment form;
- construction and analysis of a formula in conjunctive normal form.

What does your analysis tell you about the existence of errors in the program? How confident can you be about the soundness of this analysis?

[20%]