CSC410

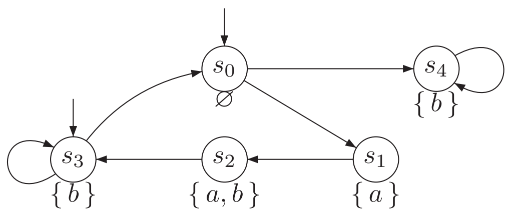
Assignment 6

Due on Monday Dec 4, 2017 before 11am

Problem 1

This problem concerns itself with CTL model checking algorithms.

(a) (6 points) Consider the transition system TS outlined below:

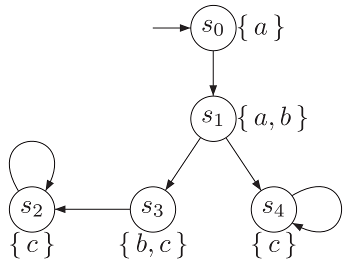


For each of the following CTL formulae determine the set $Sat(\Phi_i)$ and whether $TS \models \Phi_i$:

- $\Phi_1 = \forall(a \,\mathsf{U}\, b) \vee \exists \bigcirc (\forall \Box b)$
- $\Phi_2 = \forall \Box \forall(a \,\mathsf{U}\, b)$
- $\Phi_3 = \forall \Box \exists \Diamond \Phi_1$

Note that for this part, we are just interested in the final $Sat(\Phi_i)$ set. You do not have to include the intermediate sets that you would use to compute the final results. Each part has 2 points, exactly because the answer is exactly one set.

(b) (10 points) Consider the two CTL formulas $\Phi_1 = \exists \Diamond \forall \Box c$ and $\Phi_2 = \forall(a \,\mathsf{U}\, \forall \Diamond c)$, and the transition system TS outlined below.



Decide whether $TS \models \Phi_i$ (for $i = 1, 2$) using the CTL model checking algorithm taught in class. Sketch the main steps; by this, we mean identifying the sub-formulas and their Sat sets. You do not have to show the detail of your work for computing individual Sat sets.

Problem 2

(10 points) Let TS be a finite transition system (over AP) without terminal states (i.e. every state has an outgoing transition), and Φ and Ψ be CTL state formulae (over AP). Prove or disprove $TS \models \exists(\Phi \,\mathsf{U}\, \Psi)$ if and only if $TS' \models \exists \Diamond \Psi$ where TS' is obtained from TS by eliminating all outgoing transitions from states s such that $s \models \Psi \vee \neg \Phi$.

Problem 3

(10 points) Which one of the following equivalences is true?

- $\forall \bigcirc \forall \Diamond \Phi \equiv \forall \Diamond \forall \bigcirc \Phi$
- $\exists \bigcirc \exists \Diamond \Phi \equiv \exists \Diamond \exists \bigcirc \Phi$
- $\forall \bigcirc \forall \Box \Phi \equiv \forall \Box \forall \bigcirc \Phi$
- $\exists \bigcirc \exists \Box \Phi \equiv \exists \Box \exists \bigcirc \Phi$

Provide counterexamples for the false ones. Formally prove one of the true ones correct (you choose which one), and justify the rest using a high level argument (to show your way of coming up to the conclusion). Feel free to use the equalities provided on the slides and/or the semantics of CTL for the proof.

Problem 4: Testing

For this problem, consider the following Java program for triangle classification.

```java
package triangle;

class Triangle {
   public static final int EQUILATERAL = 0;
   public static final int INVALID = -1;
   public static final int ISOSCELES = 1;
   public static final int SCALENE = 2;

   public static void test (int x) {
     int type = classify (3, 4, x);
   }

   public static int classify (int a, int b, int c) {

     if (a <= 0 || b <= 0 || c <= 0)
       return INVALID;

     if (a==b && b==c)
       return EQUILATERAL;

     else if (a >= (b+c) || c >= (b+a) || b >= (a+c) )
       return INVALID;

     else if ((a == b && b != c ) || (a != b && c == a) || (c == b && c != a))
       return ISOSCELES;

     return SCALENE;
   }
}
```

(a) (8 points) Draw the symbolic execution tree for the `classify` method, where a, b and c are the input variables.

(b) (8 points) Illustrate how the concolic execution technique works if applied to the `classify` method. Assume the initial input is $(0, 0, 0)$. Make sure that you achieve full branch coverage.

(c) (7 points) Use SPF to generate test cases for the `test` method with `classify` inlined (you do not need to treat the function call in a special way) – treating c as symbolic variables while treating a and b as concrete variables (with value 3 and 4, respectively). Write down the test cases in a table with two columns for the value of x, and the expected return. Which branches are covered and what is the branch coverage?

(d) (7 points) Use SPF to generate test cases for the `classify` method – treating a, b, and c all as symbolic variables. Write down the test cases in a table with four columns for the values of a, b, c, and the expected return. What additional tests (if needed) are required to achieve MC/DC coverage?

Deliverables

(a) Each node in your symbolic execution tree is required to have 3 pieces of information in it: (1) path constraints collected, (2) current state (values of variables a, b, and c), and (3) next statement in line for execution.

(b) Show the execution of the program by providing step by step values for all program variables (in a vertical list) as well as the set of path constraints collected up to that point in execution. Clearly separate different executions from each other, and briefly justify the move from one execution to the next.

(c) Write down the generated test cases in a table. Briefly discuss how you compute the coverage. Submit your JPF configuration file for this test under the name `triangleC.jpf` to show your work.

(d) Write down the test cases in a table. Clearly separate the ones generated by SPF and the ones added by yourself. Briefly explain why MC/DC coverage is achieved with your set of tests. Submit your JPF configuration file for this test under the name `triangleD.jpf` to show your work.

Hint

To speed up the test generation in SPF, you can set a bound for integers between -11 to 10.