INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Date: ……….*FN / AN* Time: *3 hrs* Full marks: 70 No. of students: 56

Spring End Semester Exams, 2014 Dept: *Computer Science & Engineering* Sub No: CS60030

M.Tech (Elective) Sub Name: ***Formal Systems***

Instructions: Questions Q1, Q2, Q3 are compulsory. Answer any one among Q4 and Q5.
Answer all parts of a question in the same place.

PART-1: *Answer all questions of this part*

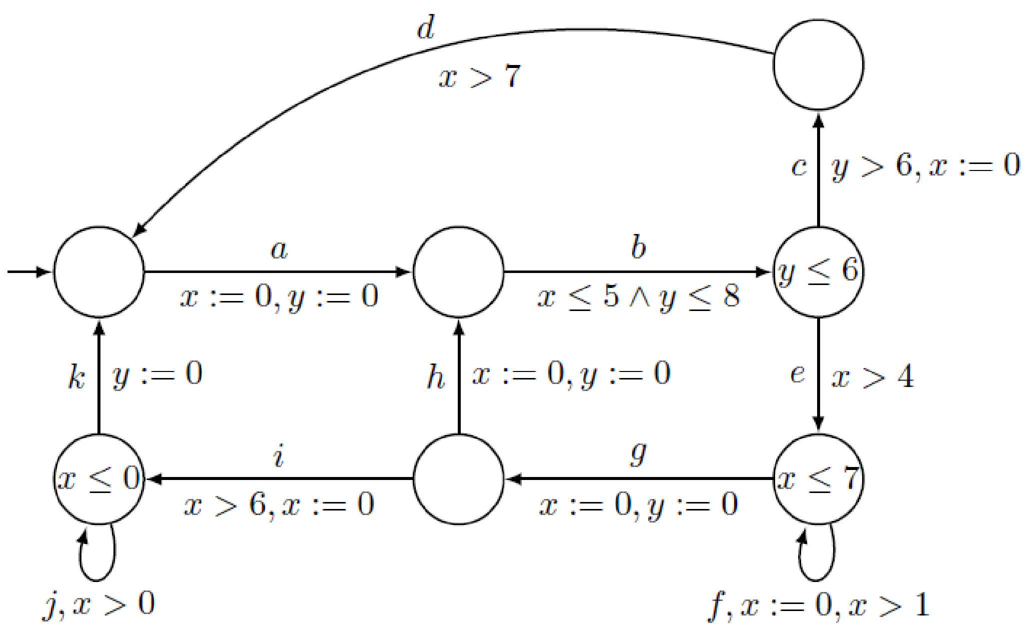
1. [Timed Systems] [4 + 3 + 3 + 10 = 20 marks]

   (a) Give the formal definition of time convergent and time divergent paths in a timed automaton.

   (b) Explain the notion of timelock in a timed automaton with a suitable example.

   (c) How can you algorithmically detect the existence of a timelock in a timed automaton? Can this be done by examining the corresponding region automaton?

   (d) Consider the following timed automata with clocks *x* and *y*, and events *a, b, c, d, e, f, g, h, i, j,* and *k.* Provide a simplified time automaton by identifying and removing redundant clock constraints and clock resets. After your simplifications, the timed behavior (the events with their time of occurrence) should remain exactly the same.

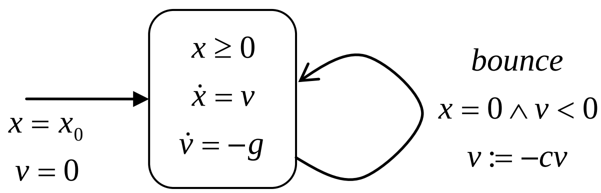


2. [Hybrid Systems] [3 + 4 + 5 + 8 = 20 marks]

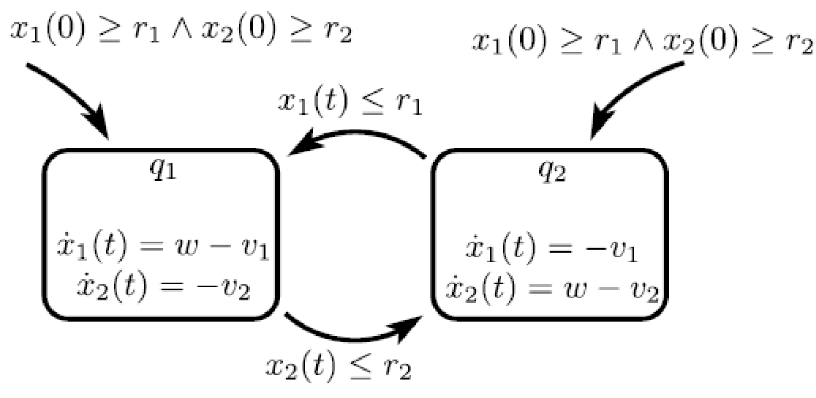
(a) Explain the different elements of a hybrid automaton, H = ⟨Loc, Var, Lab, Edg, Act, Inv⟩, using an example of your choice.

(b) Explain the convex hull operator and its use in reachability analysis of hybrid automata.

(c) Recall the hybrid automaton (given below) for a bouncing ball, where the coefficient of restitution is *c*. Let us now consider the aerodynamic drag (resistance due to air) as a force *F* which always acts in a direction opposite to the movement of the ball. Modify the given automaton to model the aerodynamic drag.



(d) Consider the following hybrid system consisting of two tanks containing water. Each tank is leaking at a constant rate. Water is added at a constant rate to the system through a hose, which at any point of time is filling either one tank or the other. It is assumed that the hose can switch between the tanks instantaneously. Let x_1 and x_2 denote the volume of water in Tank-1 and Tank-2 respectively. Let v_1 and v_2 denote the constant flow of water out of Tank-1 and Tank-2 respectively. Let *w* denote the constant flow of water into the system. The objective is to keep the water volumes above r_1 and r_2 respectively, assuming that the water volumes are above r_1 and r_2 initially. This is achieved by a controller that switches the inflow to Tank-1 whenever $x_1(t) \leq r_1$ and to Tank-2 whenever $x_2(t) \leq r_2$.



(i) Give an example of zeno behavior admitted by this automaton. Are these zeno behaviors reachable from an initial state of the system regardless of the values of v_1, v_2, r_1, r_2, w? If not, under what constraints does the zeno behavior become unreachable?

(ii) Modify the automaton to eliminate zeno behaviors. Also add the necessary location invariants which are missing in the given hybrid automaton.

3. [Program Analysis] [5 + 5 + (4+2+4) = 20 marks]

(a) Outline the method of counter-example guided abstraction refinement. Why do we need abstraction? Why do we need refinement?

(b) Explain how predicate abstraction works within the framework of counter-example guided abstraction refinement, with a suitable example.

(c) Consider the following program fragment and answer the questions that follow:

```
L0: a := 0; b:= 0; i := 0;
L1: while (a < 5000) do
L2: a := b + 1;
L3: b := a + 1;
L4: i := i + 1;
L5: // end-of-while loop
```

We wish to analyze this program using the interval abstract domain. Thus each abstract state of the variables is a triple $(I_a; I_b; I_i)$, where I_x denotes the interval of variable x. Each I_x is either $(-\infty, x_u]$, $[x_l, \infty)$, $[x_l, x_u]$, or $(-\infty, \infty)$, where x_l and x_u are integers with $x_l \le x_u$.

(i) If $(I_a; I_b; I_i)$ is the abstract state of the variables at location L1 in some iteration, then what will be the abstract state of the variables when the control returns to L1 after executing each of the statements at L2, L3, L4 once? You should give the lower and upper bounds of each variable after executing the statement at L4 in terms of the lower and upper bounds of variables just before the loop body was entered at.

(ii) How many iterations will it take to determine the range of values reachable at L1?

(iii) Recall that the standard widening operator ∇ over intervals is defined as follows:

$[x, y] = [a, b] \nabla [a', b']$,

where $x = a$ if $a \le a'$, and $x = -\infty$ otherwise, and
$y = b$ if $b' \le b$, and $y = +\infty$ otherwise

Demonstrate the use of the widening operator to approximate the range of values reachable at L1. How many iterations will it take with this operator?

PART-2: *Answer any one question from this part*

4. [Omega Regular Languages] [5 + 5 = 10 marks]

(a) Prove or disprove the following equivalences for ω-regular expressions, where E, F, F_1, F_2, denote regular expressions with $\varepsilon \notin \mathcal{L}(F) \cup \mathcal{L}(F_1) \cup \mathcal{L}(F_2)$

(i) $E.(F_1 + F_2)^\omega = E.(F_1)^\omega + E.(F_2)^\omega$

(ii) $(E^*.F)^\omega = E^*.(F)^\omega$

(b) Consider the LTL formula φ = a U (O a) over the set of atomic propositions *AP* = { a }. Construct an equivalent GNBA $\mathcal{G}$ (that is, $\mathcal{L}_\omega(\mathcal{G})$ = Words(φ)) according to the algorithm discussed in the class.

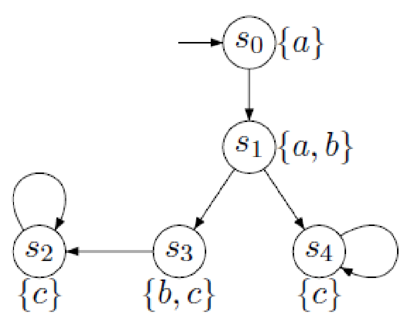
5. [Computation Tree Logic] [3 + 4 + 3 = 10 marks]

(a) Provide two transition systems TS_1 and TS_2 (without terminal states, and over the same set of atomic propositions) and a CTL formula φ, such that Traces(TS_1) = Traces(TS_2) and $TS_1 \models \varphi$, but $TS_2 \not\models \varphi$.

(b) Use the CTL model checking algorithm discussed in the class to decide whether $TS \models \varphi_1$ and whether $TS \models \varphi_2$. For each state sub-formulae, write down the set of states satisfying those formulae.

$\varphi_1 = \exists\Diamond\forall\Box c$

$\varphi_2 = \forall(a \; \mathrm{U} \; \forall\Diamond c)$



(c) Present the formal syntax of TCTL (Timed Computation Tree Logic). Illustrate each TCTL operator with a suitable example.