Automated Verification

Assignment 1

(Due on Wed 25 January 2006)

1. Recall the two versions of transition systems defined in class. The transition-labeled one was of the form $(Q, s, \longrightarrow)$ with a single initial state s and $\longrightarrow \subseteq Q \times \Sigma \times Q$. The state-labeled one was of the form $(Q, S, \longrightarrow, l)$, with a set of initial states S, $\longrightarrow \subseteq Q \times Q$, and a labeling function $l : Q \longrightarrow \Sigma$. The languages generated by these transition systems are defined in the expected way.

   (a) Prove that the class of languages defined by state-labeled and transition-labeled transition systems coincide.

   (b) Prove that the class of languages defined by transition-labeled transition systems is precisely the class of prefix-closed regular languages. (A language $L \subseteq \Sigma^*$ is *prefix-closed* if whenever a string is in L, all its prefixes are also in L.)

2. How will you check if a given (transition-labeled) transition system accepts an infinite language or not? What is the complexity of your algorithm?

3. A certain system which (among other jobs it does) produces a sequence of bits to communicate some information. These bits are meant to satisfy "even" parity in every block of 4 bits. Thus if $b_0 b_1 \cdots$ is the sequence output, then the number of 1's in $b_0 b_1 b_2 b_3$ should be even; and similarly for the block $b_4 b_5 b_6 b_7$, etc.

   Write a specification as an automaton which can check this property of the system. Assume the states of the system are labeled by the bit it emits when it does emit one, and by ϵ otherwise.

4. Recall the simple mutual exclusion protocol done in class:

Process 1	Process 2
repeat forever {	repeat forever {
l_0: /* do other jobs */	..
l_1: while (turn != 1) {	..
/* do nothing */	..
}	..
l_2: enter Critical Section	..
l_3: exit Critical Section	..
l_4: turn := 2;	
}	}

Model the system as a state-labeled transition system with the objective of checking whether it satisfies the mutual exclusion property. Assume that the initial value of the shared variable "turn" is 1. Does it satisfy the property?

5. Recall the puzzle of the man, goat, lion and cabbage. A man wants to transport his three possessions to the other side of a river bank. He has a boat in which he can take one of them at a time. The problem is that he can't leave some of them unattended to. The lion would eat the goat if he gets a chance, and similarly the goat would eat up the cabbage.

   Model the system as a transition system with a view to checking if there is a solution to the man's problem. Give a specification of *undesirable* behaviours as an automaton, so that a solution to the model-checking problem (i.e. whether the language of system model has nothing in common with the specification language) gives an answer to the man's problem.