Exercises to accompany the Part II course

Temporal Logic and Model Checking

Warning: The questions below have not been fully debugged so may contain errors. Please email `Mike.Gordon@cl.cam.ac.uk` if you think you have found any.

Eventually it is hoped to make solution notes available ... but these still have to be written.

Q1

This question concerns the software example `DIV`.

```
0:  R:=X;
1:  Q:=0;
2:  WHILE Y≤R DO
3:  (R:=R-Y;
4:   Q:=Q+1)
5:
```

Write down a total relation $\hat{R}_{\texttt{DIV}}$ that agrees with the relation $R_{\texttt{DIV}}$ given in the slides where $R_{\texttt{DIV}}$ is defined. If $R_{\texttt{DIV}}$ specifies no successor to state s then what is the successor to s specified by $\hat{R}_{\texttt{DIV}}$?

Q2

This question concerns the software example `DIV`.

(a) Write down an atomic property that expresses "when `DIV` has halted $r < q$".

(b) Compute the set of states $\{(pc, x, y, r, q) \mid R^*_{\texttt{DIV}}\ (0, 7, 2, r_0, q_0)\ (pc, x, y, r, q)\}$.

Does the atomic property (a) hold of all states in the set computed for (b)?

Q3

Modify the definition of $\mathsf{Path}\ R\ s\ \pi$ given in the slides to work when the transition relation R is represented as a set of pairs of states, $R \subseteq S \times S$, rather than as a function $R : S \to S \to \mathbb{B}$ (as is done in the slides).

Q4

Define a model M and atomic property ϕ such that $M \models \mathbf{GA}\phi$ represents the property: if `DIV` is run in a state satisfying atomic property P and if it terminates, then in the state in which it terminates atomic property Q holds (this is the partial correctness $\{P\}\texttt{DIV}\{Q\}$ in Hoare logic notation).

Can you represent the property that `DIV` always terminates when started in a state satisfying P in the form $M \models \mathbf{GA}\phi$, for suitable ϕ? Justify your answer.

Q5

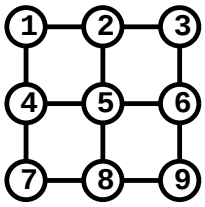
Disjunctive partitioning can sometimes be used to avoid having to build the BDD of a transition relation when symbolically computing the set of reachable states.

Explain how it might also be used to avoid building the BDD of the transition relation when generating traces to counterexamples. Illustrate you answer using the transition relation R defined by:

$$
\begin{array}{l}
R\ (x,y,z)\ (x',y',z') = \\
\quad (x' = \delta_x(x,y,z) \quad \wedge \quad y' = y \quad \wedge \quad z' = z)\ \vee \\
\quad (x' = x \quad \wedge \quad y' = \delta_y(x,y,z) \quad \wedge \quad z' = z)\ \vee \\
\quad (x' = x \quad \wedge \quad y' = y \quad \wedge \quad z' = \delta_z(x,y,z))
\end{array}
$$

Q6

This questions concerns the nine switches puzzle in the slides:

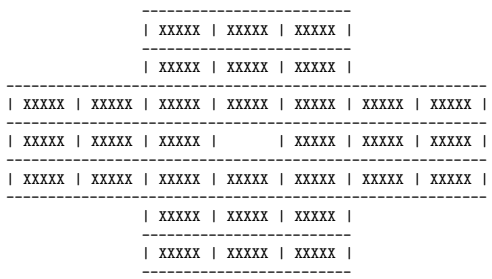


By defining a state transition function δ_i for each switch ($1 \leq i \leq 9$) express the transition relation `Trans` as the asynchronous interleaving semantics of nine state machines in parallel.

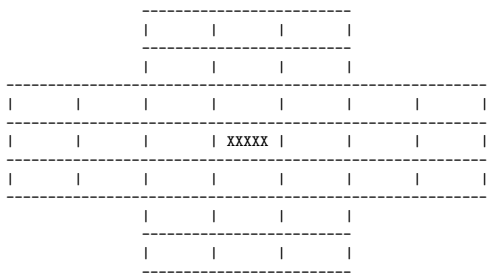
Comment on how this might help with solving the problem by symbolic model checking.

Q7

Consider the following board which is meant to represent the initial state of the puzzle Peg Solitaire.



All the positions in the board, except the one in the middle, are occupied by pegs, denoted by xxxxx. A move consists of 'jumping' a peg over an adjacent peg in the same row or column into a hole, and removing the peg that was jumped over from the board (thereby reducing the number of pegs on the board by one). The puzzle is to find a sequence of moves, starting from the above configuration, to a configuration consisting of just one peg in the middle, i.e.:



Describe how you could formulate Peg Solitaire as the problem of computing the set of reachable states.

Would disjunctive partitioning be useful?

Hint: Your answers to the previous two questions might be useful.

Q8

This question concerns the 2-thread program JM1 described in the slides.

Thread 1	Thread 2
0: IF LOCK=0 THEN LOCK:=1;	0: IF LOCK=0 THEN LOCK:=1;
1: X:=1;	1: X:=2;
2: IF LOCK=1 THEN LOCK:=0;	2: IF LOCK=1 THEN LOCK:=0;
3:	3:

Draw the computation tree of states $(pc_1, pc_2, lock, x)$ of JM1 starting at state $(0,0,0,0)$ (i.e. $pc_1 = 0 \wedge pc_2 = 0 \wedge lock = 0 \wedge x = 0$).

Let $M_{\texttt{JM1}} = (S_{\texttt{JM1}}, \{(0,0,0,0)\}, R_{\texttt{JM1}}, AP)$, where $R_{\texttt{JM1}}$ is a total[1] transition relation corresponding to your computation tree and AP contains all atomic properties of the form $\langle x{=}v\rangle$ which mean state component x has value v (e.g. $\langle lock{=}0\rangle$ means $(\lambda(pc_1, pc_2, lock, x).\ lock = 0)$).

Explain the meaning of each of the following LTL properties and say whether it is true.

$M_{\texttt{JM1}} \models \mathbf{F}\langle pc_1{=}3\rangle$
$M_{\texttt{JM1}} \models \mathbf{G}(\langle lock{=}1\rangle \Rightarrow \mathbf{F}\langle lock{=}0\rangle)$
$M_{\texttt{JM1}} \models \mathbf{G}(\langle pc_1{=}2\rangle \Rightarrow \mathbf{X}\langle pc_1{=}3\rangle)$
$M_{\texttt{JM1}} \models \mathbf{F}(\langle pc_1{=}1\rangle \wedge \langle pc_2{=}1\rangle)$
$M_{\texttt{JM1}} \models \mathbf{G}(\langle pc_1{=}3\rangle \Rightarrow \mathbf{G}\langle pc_1{=}3\rangle)$

Explain the meaning of each of the following CTL properties and say whether it is true.

$M_{\texttt{JM1}} \models \mathbf{EF}\langle pc_1{=}3\rangle$
$M_{\texttt{JM1}} \models \mathbf{EFAF}\langle x{=}1\rangle$
$M_{\texttt{JM1}} \models \mathbf{EF}(\langle lock{=}0\rangle \wedge \langle x{=}1\rangle)$
$M_{\texttt{JM1}} \models \mathbf{E}[\langle lock{=}0\rangle \mathbf{U} \langle x{=}2\rangle]$

Explain the meaning of each of the following CTL* properties and say whether it is true.

$M_{\texttt{JM1}} \models \mathbf{A}(\mathbf{FG}\langle lock{=}0\rangle \vee \mathbf{F}\langle x{=}2\rangle)$
$M_{\texttt{JM1}} \models \mathbf{E}(\mathbf{X}\langle pc_1{=}1\rangle \wedge \mathbf{F}\langle pc_1{=}3\rangle)$
$M_{\texttt{JM1}} \models \mathbf{A}(\mathbf{X}\langle pc_1{=}1\rangle \Rightarrow \mathbf{F}\langle pc_1{=}3\rangle)$
$M_{\texttt{JM1}} \models \mathbf{A}(\mathbf{G}(\langle pc_1{=}1\rangle \Rightarrow \mathbf{X}(\mathbf{G}\langle x{=}1\rangle)))$

[1] **Hint:** see Q1.

Q9

This question uses the 2-thread program JM1 described in the slides (and also used in the preceding question).

In the slide on model checking $\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]$ the set of marked states $\{\!\{\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]\}\!\}$ is defined by:

$$\{\!\{\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]\}\!\} = \bigcup_{n=0}^{\infty} \{\!\{\mathbf{E}[\psi_1\ \mathbf{U}\ \psi_2]\}\!\}_n$$

For the model $M_{\texttt{JM1}}$, calculate $\{\!\{\mathbf{E}[\langle lock{=}0\rangle \mathbf{U} \langle x{=}2\rangle]\}\!\}$ and $\{\!\{\mathbf{E}[\langle pc_1{=}0\rangle \mathbf{U} \langle x{=}2\rangle]\}\!\}$ and explain how this is used to check:

$$M_{\texttt{JM1}} \models \mathbf{E}[\langle lock{=}0\rangle \mathbf{U} \langle x{=}2\rangle]$$
$$M_{\texttt{JM1}} \models \mathbf{E}[\langle pc_1{=}0\rangle \mathbf{U} \langle x{=}2\rangle]$$

Are either of these true? Explain your answer.

Q10

Consider the program DIV used in the slides:

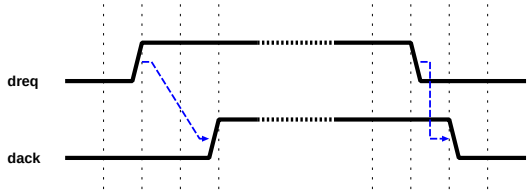
```
0:  R:=X;
1:  Q:=0;
2:  WHILE Y≤R DO
3:  (R:=R-Y;
4:   Q:=Q+1)
5:
```

Suppose the program variables X, Y, Q, R are restricted to natural numbers less than 256.

Explain how you might represent sets of states and the transition relation as BDDs suitable for use in symbolic model checking. You need not give full details, but should describe how such details would be generated.

Q11

The timing diagram below is mentioned in the slides.



The following two handshake properties were given:

- following a rising edge on `dreq`, the value of `dreq` remains 1 (i.e. *true*) until it is acknowledged by a rising edge on `dack`
- following a falling edge on `dreq`, the value on `dreq` remains 0 (i.e. *false*) until the value of `dack` is 0

Formalise these two properties as formulae in a suitable temporal logic. You should state what logic you are using and briefly describe why you chose it.

Q12

The `DIV` example is discussed in the slides:

```
0:  R:=X;
1:  Q:=0;
2:  WHILE Y≤R DO
3:  (R:=R-Y;
4:   Q:=Q+1)
5:
```

$$
\begin{array}{lcl}
\texttt{AtStart}\ (pc, x, y, r, q) & = & (pc = 0) \\
\texttt{AtEnd}\ (pc, x, y, r, q) & = & (pc = 5) \\
\texttt{InLoop}\ (pc, x, y, r, q) & = & (pc \in \{3, 4\}) \\
\texttt{YleqR}\ (pc, x, y, r, q) & = & (y \leq r) \\
\texttt{Invariant}\ (pc, x, y, r, q) & = & (x = r + (y \times q))
\end{array}
$$

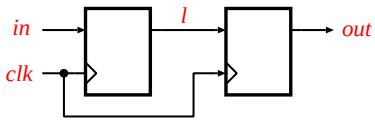
The following three properties were given:

- on every execution if `AtEnd` is true then `Invariant` is true and `YleqR` is not true
- on every execution there is a state where `AtEnd` it true
- on any execution if there exists a state where `YleqR` is true then there is also a state where `InLoop` is true

Formalise these three properties as formulae in a suitable temporal logic. You should state what logic you are using and briefly describe why you chose it.

Q13

The following circuit consisting of two Dtype in series is mentioned in the slides.



The behaviour of this was described informally the giving the trace:

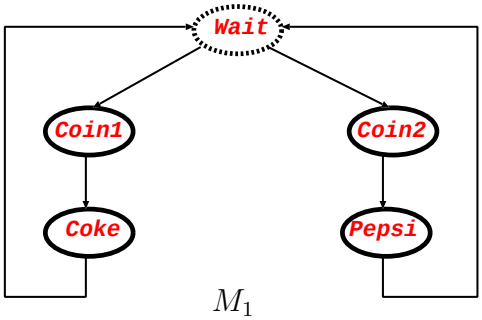
in aaaaaaaaaaabbbbbbccccccddddddddd.....
clk 000001111100000111110000011111100.....
l eeeeeaaaaaaaaaabbbbbbbbbbddddddd.....
out fffffeeeeeeeeeeaaaaaaaaaabbbbbbb.....

Call this example `D2`. Devise a model to represent `D2` suitable for use in model checking.

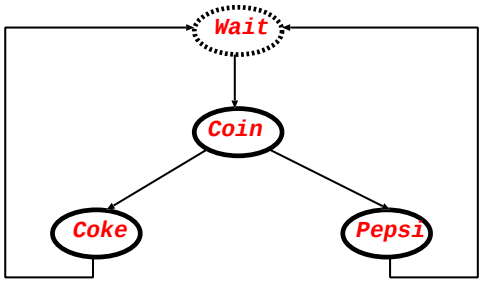
Hint: include in the state components for both the current value of *clk* and for its value at the previous time instant, say clk^- (e.g. take the state to be (clk^-, clk, in, l, out)).

Q14

Let models M_1 and M_2 correspond to the state transition diagrams below. Assume $AP = \{Wait, Coin1, Coin2, Coin, Coke, Pepsi\}$. Initial states are indicated by a dotted line, and state names are also used to name atomic predicates that only hold at the state with the same name. $Coin$ holds of two states: $Coin \in L_{M_1}(Coin1)$ and $Coin \in L_{M_1}(Coin2)$.



M_1



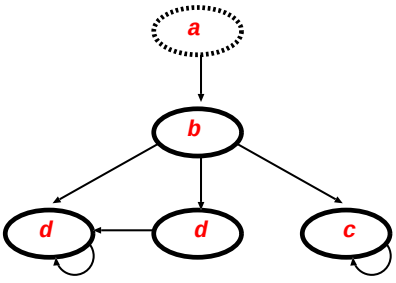
M_2

Are M_1 and M_2 bisimilar, i.e. $M_1 \equiv M_2$? Justify your answer.

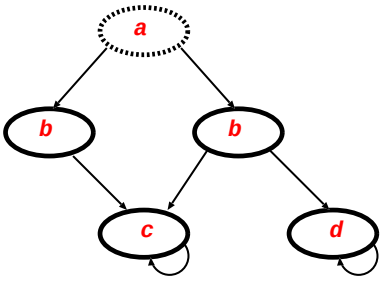
Hint: consider $\mathbf{AG}(Coin \Rightarrow \mathbf{EX}Coke)$.

Q15

Define models M_1 and M_2 that correspond to the state transition diagrams below. The initial states are indicated by a dotted line, and the atomic predicates are shown inside the states where they hold.



M_1



M_2

Does $M_1 \preceq M_2$ or $M_2 \preceq M_1$? Justify your answer. Does it shed light on whether $M_1 \preceq M_2$ and $M_2 \preceq M_1$ entails $M_1 \equiv M_2$?