Model Checking – Exercise sheet 5

Exercise 5.1
Consider the following NBA with the acceptance set \( F = \{ s_1, s_6 \} \). Apply the nested depth-first search approach to verify that \( L(A) \neq \phi \).

![NBA Diagram]

Exercise 5.2
Consider the Promela model below which addresses the mutual exclusion problem by using a semaphore \( s \). When \( s \) is false, a process may enter its critical section and set \( s \) to true. The semaphore is reset to false when the process leaves its critical section.

```promela
bool s;
active [2] proctype m() {
    idle:
    skip;
    wait:
        atomic { (!s) -> s = true; }
    cs:
        s = false;
    goto idle;
}
```

We consider the following properties:

a) Both processes cannot enter the critical section at the same time.

b) Whenever a process waits, it will eventually enter the critical section.

Follow step-by-step the outline given below to model check the properties:
(i) Construct a state transition system from the model.

(ii) Write down an atomic proposition $AP$ and an LTL formula $\phi$ for each properties.

(iii) Construct a Büchi automaton $B_{\neg\phi}$ for the negation of the formula $\phi$.

(iv) Construct from the transition system the Kripke structure $K$ and the Büchi automaton $B_K$ over $AP$.

(v) Construct the intersection Büchi automaton $B$ for $B_K$ and $B_{\neg\phi}$.

(vi) Run the emptiness algorithm in the lecture to check whether $L(B) = \emptyset$:

- If $L(B) = \emptyset$, the property holds, i.e. $K \models \phi$.
- If $L(B) \neq \emptyset$, the property does not hold, i.e. $K \not\models \phi$.

   In this case, find a counterexample run that violates the property. How to obtain a counterexample in general?

(vii) Use Spin to confirm your results.

First do step (i), and then steps (ii)-(vii) separately for each property (a) and (b). Write down all intermediary results.
Solution 5.1
Let’s assume that the algorithm selects the next state from top-to-bottom and left-to-right.
It first finds the SCC \{s_2, s_3\} which does not have any accepting state. Then it goes to state \(s_4\) and later finds the SCC \{s_4, s_5, s_6\} when it reaches \(s_6\). Since \(s_6\) is an accepting state, the algorithm terminates and reports that the language of the NBA is non-empty.

Solution 5.2
(i) \(\mathcal{T} = (S, \to, r)\), where \(S = \{i_0, w_0, cs_0\} \times \{i_1, w_1, cs_1\} \times \{t, f\}\) for modeling three locations in \(m\) and two possible values of \(s\).

(ii) \(\phi_a = \neg F(cs_0 \land cs_1)\), where \(AP_a = \{cs_0, cs_1\}\) and 
\(\phi_b = G(w_0 \rightarrow F cs_0)\), where \(AP_b = \{w_0, cs_0\}\)

(iii) \(\neg \phi_a = F(cs_1 \land cs_2)\). So, \(\mathcal{B}_{\neg \phi_a}\) can be constructed as follows:

\(\neg \phi_b = F(w_0 \land G \neg cs_0)\). So, \(\mathcal{B}_{\neg \phi_b}\) can be constructed as follows:
(iv) Let rename the states to $S = \{s_0, \ldots, s_7\}$.

$K_a = (S, \rightarrow, r, AP_a, \nu_a)$, where $\nu_a(s) = \begin{cases} \{cs_0\}, & \text{if } s \in \{s_3, s_6\} \\ \{cs_1\}, & \text{if } s \in \{s_5, s_7\} \\ \emptyset, & \text{otherwise} \end{cases}$

$K_b = (S, \rightarrow, r, AP_b, \nu_b)$, where $\nu_b(s) = \begin{cases} \{w_0\}, & \text{if } s \in \{s_1, s_4, s_7\} \\ \{cs_0\}, & \text{if } s \in \{s_3, s_6\} \\ \emptyset, & \text{otherwise} \end{cases}$

The Büchi automaton $B_{K_a} = (2^{AP_a}, S, r, \Delta_{K_a}, S)$, where $\Delta_{K_a} = \{(s, \nu_a(s), t) | s \rightarrow t\}$: 
The Büchi automaton $B_{K_b} = (2^{AP_b}, S, r, \Delta_{K_b}, S)$, where $\Delta_{K_b} = \{(s, \nu_b(s), t) \mid s \rightarrow t\}$:

(v) $B_a = (2^{AP_a}, S \times \{q_0, q_1\}, (s_0, q_0), \Delta_a, S \times \{q_1\})$
\( B_b = (2^{2^{\mathcal{A}_b}}, S \times \{q_0, q_1\}, (s_0, q_0), \Delta_b, S \times \{q_1\}) \)

(vi) For \( B_a \), the algorithm terminates without reporting a counterexample. By the time the algorithm terminates, it finds out that every state forms an SCC, but without an accepting state.

For \( B_b \), assuming that the algorithm always searches the automaton above top-to-bottom and left-to-right, then it first finds the SCC \( \{(s_0, q_0), (s_1, q_0), (s_3, q_0)\} \) when it reaches \((s_3, q_0)\). Later, it finds the SCC \( \{(s_6, q_0), (s_2, q_0), (s_4, q_0)\} \) when it reaches \((s_4, q_0)\). Finally, it reaches \((s_4, q_1)\) and finds out that its successor \((s_7, q_1)\) is still active. Since \((s_4, q_1)\) is an accepting state, the algorithm stops. Notice that the last SCC it discovers is \( \{(s_7, q_1), (s_1, q_1), (s_4, q_1)\} \) with \((s_7, q_1)\) as the root.

The counterexample run in \( B_b \) found by the algorithm is

\[(s_0, q_0), (s_1, q_0), (s_3, q_0), (s_6, q_0), (s_2, q_0), (s_4, q_0), ((s_7, q_1), (s_1, q_1), (s_4, q_1))^{\omega}\]

The corresponding path in \( K_b \) can be obtained by projecting the run on the first component. The counterexample valuation sequence is as follows:

\[\{\} \{w_0\} \{c_s_0\} \{c_s_0\} \{\} \{w_0\} \{w_0\} \{w_0\}^{\omega}\]

(vii) Check the formulas: \(!<> \!(m[0]@cs \& \& m[1]@cs)\) and \([](m@wait -> <> m@cs)\)