Discrete Event Systems
Solution to Exercise Sheet 10

1 Comparison of Finite Automata

Here are two simple finite automata:

For each, we have a one bit encoding for the states ($x_A$ and $x_B$), one binary output ($y_A$ and $y_B$), and one common binary input ($u$). We want to verify whether or not these two automata are equivalent. This can be done through the following steps:

a) Express the characteristic function of the transition relation for both automaton, $\psi_r(x, x', u)$.

b) Express the joint transition function, $\psi_f$.

Reminder: $\psi_f(x_A, x'_A, x_B, x'_B) = (\exists u: \psi_A(x_A, x'_A, u) \cdot \psi_B(x_B, x'_B, u))$.

c) Express the characteristic function of the reachable states, $\psi_X(x_A, x_B)$.

d) Express the characteristic function of the reachable output, $\psi_Y(y_A, y_B)$.

e) Are the two automata equivalent? Hint: Evaluate, for example, $\psi_Y(0, 1)$.

\[ \psi_A(x_A, x'_A, u) = x_A x'_A \cdot u + x_A x'_B u + x_A x'_A u + x_A x_A u \]
\[ \psi_B(x_B, x'_B, u) = x_B x'_B \cdot u + x_B x'_B u + x_B x'_B u + x_B x'_B u \]

\[ \psi_f(x_A, x'_A, x_B, x'_B) = (x_A x'_A + x_B x'_B + x_B x'_B + x_A x'_B) \cdot (x_B x'_B + x_A x'_B) + (x_A x'_A + x_A x'_B) \cdot (x_B x'_B + x_A x'_B) \]
\[ = x_A x'_A x_B + x_A x'_A x_B + x_A x'_A x_B + x_A x'_A x_B + x_A x'_A x_B + x_A x'_A x_B + x_A x'_A x_B + x_A x'_A x_B \]

\[ \psi_Y(y_A, y_B) = \]

\[ \psi_Y(0, 1) \]

\[ \psi_Y(0, 1) \]

\[ \psi_Y(0, 1) \]

\[ \psi_Y(0, 1) \]

\[ \psi_Y(0, 1) \]
\[ \psi_X(x'_A, x'_B) = \psi_X(x'_A, x'_B) + (\exists (x_A, x_B) : \psi_X(x_A, x_B) \cdot \psi_f(x_A, x'_A, x'_B)) \]
\[ = x_A' x'_B + x_A x'_B + x'_A x_B \]
\[ \psi_{X_2}(x'_A, x'_B) = x_A' x'_B + x_A x'_B + x_A x'_B + x_A x_B = \psi_{X_2} \quad \text{⇒ the fix-point is reached!} \]
\[ \Rightarrow \psi_X(x_A, x_B) = x_A x_B + x_A x_B + x_A x_B + x_A x_B \]
\[ \text{d) Here you first need to express the output function of each automaton, that is the feasible combinations of states and outputs,} \]
\[ \psi_{g_A} = x_A y_A + x_A y_A \quad \text{and} \quad \psi_{g_B} = x_B y_B + x_B y_B \]
Then the reachable outputs are the combination of the reachable states and the outputs functions, that is,
\[ \psi_Y(y_A, y_B) = (\exists (x_A, x_B) : \psi_X \cdot \psi_{g_A} \cdot \psi_{g_B}) \]
\[ = y_A y_B + y_A y_B + y_A y_B + y_A y_B \]
\[ \text{e) From the reachable output function, we see that these automata are not equivalent. Indeed, there exists a reachable output admissible (} \psi_Y((y_A, y_B) = (0,1)) = 1 \text{) for which} y_A \neq y_B. \]
Another way of saying looking at it: \[ \psi_Y \cdot (y_A \neq y_B) \neq 0, \]
where \((y_A \neq y_B) = y_A y_B + y_A y_B. \]
2 Temporal Logic

a) We consider the following automaton. The property \( a \) is true on the colored states (0 and 3).

![Automaton Diagram]

For each of the following CTL formula, list all the states for which it holds true.

(i) \( \text{EF } a \)
(ii) \( \text{EG } a \)
(iii) \( \text{EX AX } a \)
(iv) \( \text{EF } (a \text{ AND EX NOT}(a)) \)

(i) \( Q = \{0, 1, 2, 3\} \)
(ii) \( Q = \{0, 3\} \)
(iii) \( \text{(AX } a \text{) holds for } \{2, 3\}, \text{ thus } Q = \{1, 2\} \)
(iv) \( \text{(a AND EX NOT}(a)) \text{ is true for states where } a \text{ is true and there exists a direct successor for which it is not. Only state 0 satisfy this (from it you can transition to 1, where } a \text{ does not hold). Moreover, state 0 is reachable for all states in this automaton ("from all states there exists a path going through 0 at some point") Hence } Q = \{0, 1, 2, 3\} \)

b) Given the transition function \( \psi_f(q, q') \) and the characteristic function \( \psi_Z(q) \) for a set \( Z \), write a small pseudo-code which returns the characteristic function of \( \psi_{AFZ}(q) \). It can be expressed as symbolic boolean functions, like \( X_A x_A x_B x_B' + X_A x_A' x_B x_B' \).

**Hint:** To do this, simply use the classic boolean operators AND, OR, NOT and \(!=\). You can also use the operator \( \text{PRE}(Q, f) \), which returns the predecessor of the set \( Q \) by the transition function \( f \). That is,

\[
\text{PRE}(Q, f) = \{q' : \exists q, \psi_f(q', q) \cdot \psi_Q(q) = 1\}
\]

**Hint:** It can be useful to reformulate \( AFZ \) as another CTL formula.

Here, the trick is to remember that \( AFZ \equiv \text{NOT}(EG \text{ NOT}(Z)) \). Hence, one can compute the function for \( EG \text{ NOT}(Z) \) quite easily (following the procedure given in the lecture) and take the negation in the end. A possible pseudo-code doing this is the following,

**Require:** \( \psi_Z, \psi_f \)

\[
\begin{align*}
\text{current} & = \text{NOT}(\psi_Z); \\
\text{next} & = \text{current AND } \psi_{\text{PRE}(\text{current}, f)}; \\
\text{while } & \text{next } \neq \text{current } \text{do} \\
& \text{current} = \text{next}; \\
& \text{next} = \text{current AND } \psi_{\text{PRE}(\text{current}, f)}; \\
\text{end while} \\
\text{return } & \psi_{AFZ} = \text{NOT}(\text{current});
\end{align*}
\]

\( \text{\triangleright } \) Equivalence in term of sets:

\[
\begin{align*}
\triangleright & \ X_0 \\
\triangleright & \ X_1 = X_0 \cap \text{Pre}(X_0, f) \\
\triangleright & \ X_i |\!\!\!\!\!\!\!\!= \text{EG NOT}(Z) \\
\triangleright & \ X_i |\!\!\!\!\!\!\!\!= \text{AF } Z = \text{NOT}(\text{EG NOT}(Z))
\end{align*}
\]