Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Engineering and Networks Laboratory
Computer Engineering Group (TEC)

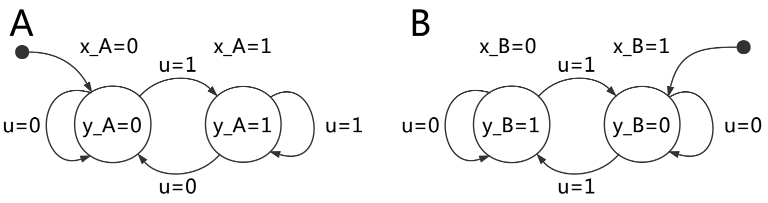
Prof. L. Thiele / X.He

Discrete Event Systems

Solution to Exercise: Verification of Finite Automata (Part 2)

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, ψ_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, $\psi_Y(0, 1)$.

a) $\psi_A(x_A, x'_A, u) = \overline{x_A}\,\overline{x'_A}\,\overline{u} + \overline{x_A}x'_A u + x_A x'_A u + x_A \overline{x'_A}\,\overline{u}$
$\psi_B(x_B, x'_B, u) = \overline{x_B}\,\overline{x'_B}\,\overline{u} + \overline{x_B}x'_B u + x_B x'_B \overline{u} + x_B \overline{x'_B} u$

b)
$$\begin{aligned}\psi_f(x_A, x'_A, x_B, x'_B) &= (\overline{x_A}x'_A + x_A x'_A) \cdot (\overline{x_B}x'_B + x_B\overline{x'_B}) + \\ &\quad (\overline{x_A}\,\overline{x'_A} + x_A\overline{x'_A}) \cdot (\overline{x_B}\,\overline{x'_B} + x_B x'_B) \\ &= \overline{x_A}x'_A\overline{x_B}x'_B + \overline{x_A}x'_A x_B\overline{x'_B} + x_A x'_A\overline{x_B}x'_B + x_A x'_A x_B\overline{x'_B} + \\ &\quad \overline{x_A}\,\overline{x'_A}\,\overline{x_B}\,\overline{x'_B} + \overline{x_A}\,\overline{x'_A}x_B x'_B + x_A\overline{x'_A}\,\overline{x_B}\,\overline{x'_B} + x_A\overline{x'_A}x_B x'_B\end{aligned}$$

c) Computation of the reachable states is performed incrementally. Starts with the initial state of the system $\psi_{X_0}(x_A, x_B) = \overline{x_A}x_B$ and then add the successors until reaching a fix-point,

$$\begin{aligned}
\psi_{X_1}(x'_A, x'_B) &= \psi_{X_0}(x'_A, x'_B) + (\exists(x_A, x_B) : \psi_{X_0}(x_A, x_B) \cdot \psi_f(x_A, x'_A, x_B, x'_B)) \\
&= \overline{x'_A}x'_B + \overline{x'_A}x'_B + x'_A\overline{x'_B} \\
&= \overline{x'_A}x'_B + x'_A\overline{x'_B} \\
\psi_{X_2}(x'_A, x'_B) &= \overline{x'_A}x'_B + x'_A\overline{x'_B} + x'_Ax'_B + \overline{x'_Ax'_B} \\
\psi_{X_3}(x'_A, x'_B) &= \overline{x'_A}x'_B + x'_A\overline{x'_B} + x'_Ax'_B + \overline{x'_Ax'_B} = \psi_{X_2} \qquad \rightarrow \text{the fix-point is reached!}
\end{aligned}$$

$$\Rightarrow \quad \boxed{\psi_X(x_A, x_B) = \overline{x_A}x_B + x_A\overline{x_B} + x_Ax_B + \overline{x_Ax_B}}$$

d) Here you first need to express the output function of each automaton, that is the feasible combinations of states and outputs,

$\psi_{g_A} = \overline{x_Ay_A} + x_Ay_A$ and $\psi_{g_B} = \overline{x_B}y_B + x_B\overline{y_B}$

Then the reachable outputs are the combination of the reachable states and the outputs functions, that is,

$$\begin{aligned}
\psi_Y(y_A, y_B) &= (\exists(x_A, x_B) : \psi_X \cdot \psi_{g_A} \cdot \psi_{g_B}) \\
&= y_Ay_B + \overline{y_Ay_B} + \overline{y_A}y_B + y_A\overline{y_B}
\end{aligned}$$

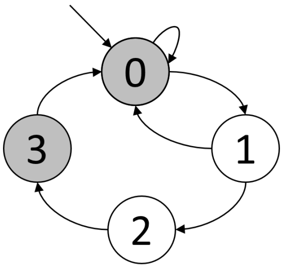
e) From the reachable output function, we see that these automata are not equivalent. Indeed, there exists a reachable output admissible $(\psi_Y((y_A, y_B) = (0, 1)) = 1)$ for which $y_A \neq y_B$.

Another way of saying looking at it: $\psi_Y \cdot (y_A \neq y_B) \neq 0$,

where $(y_A \neq y_B) = \overline{y_A}y_B + y_A\overline{y_B}$.

2 Temporal Logic

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



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

(i) EF a

(ii) EG a

(iii) EX AX a

(iv) EF (a `AND` EX `NOT`(a))

(i) $Q = \{0, 1, 2, 3\}$

(ii) $Q = \{0, 3\}$

(iii) (AX a) holds for $\{2, 3\}$, thus $Q = \{1, 2\}$

(iv) (a `AND` EX `NOT`(a)) is true for states where a 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 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 $\psi_{\text{AF } Z}(q)$. It can be expressed as symbolic boolean functions, like $\overline{x_A}x'_A\overline{x_B}x'_B + \overline{x_A}x'_Ax_Bx'_B$.
Hint: To do this, simply use the classic boolean operators `AND`, `OR`, `NOT` and $! =$. You can also use the operator `PRE`(Q, f), which returns the predecessor of the set Q by the transition function f. That is,

$$\texttt{PRE}(Q, f) = \{q' : \exists q,\ \psi_f(q', q) \cdot \psi_Q(q) = 1\}$$

Hint: It can be useful to reformulate AFZ as another CTL formula.

Here, the trick is to remember that AF $Z \equiv$ `NOT`(EG `NOT`(Z)). Hence, one can compute the function for EG `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: $\psi_Z$, $\psi_f$                                   ▷ Equivalence in term of sets:
  current = NOT($\psi_Z$);                                                    ▷ $X_0$
  next = current AND $\psi_{\texttt{PRE}(current,f)}$;                     ▷ $X_1 = X_0 \cap Pre(X_0, f)$
  while next != current do                                          ▷ $X_i != X_{i-1}$
      current = next;
      next = current AND $\psi_{\texttt{PRE}(current,f)}$;            ▷ $X_i = X_{i-1} \cap Pre(X_{i-1}, f)$
  end while                                                    ▷ $X_f \models$ EG NOT($Z$)
  return $\psi_{\text{AF } Z}$ =NOT(current);          ▷ $\overline{X_f} \models$ AF $Z$ = NOT(EG NOT($Z$))
```