Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Networked Systems Group (NSG)

HS 2021

R. Jacob / T. Schneider
based on Prof. L. Thiele's material

Discrete Event Systems

Solution to Exercise Sheet 11

1 Temporal Logic

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) (i) $\neg \text{AF}\, Z \equiv \text{EG}\, \neg Z$

(ii) We will first compute the function $Q_k \models \text{EG}\, \neg Z$, which we can compute quite easily (following the procedure given in the lecture), and take the negation in the end.

$$\begin{aligned} Q_0 &= S \backslash Z \\ Q_{i+1} &= Q_i \cap Pre(Q_i, f) \\ k &= \min\{i \mid Q_{i+1} = Q_i\} \\ Q_{\text{AF } Z} &= Z \backslash Q_k \end{aligned}$$

The main idea is that we start with the states that are not in Z. Then, at each iteration, we create an intersection between the current set of states, and all predecessors from which we can reach one of the states in the set. By doing this, we will remove any states from which there exists some future, in which Z does not hold. We stop the iteration once nothing changes anymore (we define k to be the first index for which the set of states remains the same). Hence, we express have $Q_k \models \text{EG}\, \neg Z$. What is left to do is to negate the final set (every state which is not present in Q_k).

(iii) We translate the procedure above directly into an algorithm:

```
Require: $\psi_Z$, $\psi_f$
  $\psi_{cur} \leftarrow \neg\psi_Z$
  $\psi_{next} \leftarrow \psi_{cur} \wedge \psi_{Pre(\psi_{cur}, f)}$
  while $\psi_{cur} \neq \psi_{next}$ do
      $\psi_{cur} \leftarrow \psi_{next}$
      $\psi_{next} \leftarrow \psi_{cur} \wedge \psi_{Pre(\psi_{cur}, f)}$
  end while
  return $\psi_{\text{AF}\, Z} = \neg\psi_{cur}$
```

2 Safe Network-Wide Configuration Updates

a) A forwarding loop exists in the following forwarding states. We only show the necessary parts, the others can be chosen arbitrarily:

(i) $\rho_1(v_0) = v_0$

(ii) $\rho_2(v_0) = v_1$, $\rho_2(v_1) = v_0$

(iii) $\rho_3(v_0) = v_1$, $\rho_3(v_1) = v_1$

(iv) $\rho_4(v_0) = v_1$, $\rho_4(v_1) = v_2$, $\rho_4(v_2) = v_0$

(v) $\rho_5(v_0) = v_1$, $\rho_5(v_1) = v_2$, $\rho_5(v_2) = v_1$

(vi) $\rho_6(v_0) = v_1$, $\rho_6(v_1) = v_2$, $\rho_6(v_2) = v_2$

(vii) $\rho_7(v_0) = v_2$, $\rho_7(v_2) = v_0$

(viii) $\rho_8(v_0) = v_2$, $\rho_8(v_2) = v_1$, $\rho_8(v_1) = v_0$

(ix) $\rho_9(v_0) = v_2$, $\rho_9(v_2) = v_1$, $\rho_9(v_1) = v_1$

(x) $\rho_{10}(v_0) = v_2$, $\rho_{10}(v_2) = v_1$, $\rho_{10}(v_1) = v_2$

(xi) $\rho_{11}(v_0) = v_2$, $\rho_{11}(v_2) = v_2$

b) $\rho(v_0) \neq t$ is the only next hop that is not allowed. Hence, we write:

$$\psi_{topo}(\mathbf{Z}) = \neg\left(z_0^1 z_0^0\right)$$

c) We start by case distinction. The first case is where v_0 routes traffic towards v_1, where v_1 either sends traffic to t either directly, or via v_2. The second case is where v_0 sends traffic towards v_2, which is to be handled symmetrically to before.

$$\begin{aligned}\psi_t(\mathbf{Z}) = & \left(\overline{z_0^1} z_0^0 \left[z_1^1 z_1^0 + \left(z_1^1 \overline{z_1^0} z_2^1 z_2^0\right)\right]\right) && // \; z_0 \rightarrow z_1 \\ & + \left(z_0^1 \overline{z_0^0} \left[\left(z_1^1 z_1^0 \overline{z_2^1} z_2^0\right) + z_2^1 z_2^0\right]\right) && // \; z_0 \rightarrow z_2\end{aligned}$$

d) We will perform a similar case distinction as in Exercise **b)**:

$$\begin{aligned}\psi_{v_2}(\mathbf{Z}) = & \; \overline{z_0^1} z_0^0 z_1^1 \overline{z_1^0} && // \; z_0 \rightarrow z_1 \\ & + z_0^1 \overline{z_0^0} && // \; z_0 \rightarrow z_2\end{aligned}$$

e) Notice, that we can simply write $\psi_\phi(\mathbf{Z}) = \psi_t(\mathbf{Z}) \cdot \psi_{v_2}(\mathbf{Z})$.

$$\begin{aligned}\psi_\phi(\mathbf{Z}) &= \psi_t(\mathbf{Z}) \cdot \psi_{v_2}(\mathbf{Z}) \\ &= \left[\left(\overline{z_0^1} z_0^0 \left[z_1^1 z_1^0 + \left(z_1^1 \overline{z_1^0} z_2^1 z_2^0\right)\right]\right) + \left(z_0^1 \overline{z_0^0} \left[\left(z_1^1 z_1^0 \overline{z_2^1} z_2^0\right) + z_2^1 z_2^0\right]\right)\right] \cdot \left[\overline{z_0^1} z_0^0 z_1^1 \overline{z_1^0} + z_0^1 \overline{z_0^0}\right] \\ &= \left(\overline{z_0^1} z_0^0 z_1^1 \overline{z_1^0} z_2^1 z_2^0\right) + \left(z_0^1 \overline{z_0^0} \left[\left(z_1^1 z_1^0 \overline{z_2^1} z_2^0\right) + z_2^1 z_2^0\right]\right)\end{aligned}$$

Another way to solve this problem is to repeat the same idea from Exercise **b)** and **c)**. In fact, there only exists three valid paths through the network, that satisfy both constraints. These are: $[v_0, v_1, v_2, t]$, $[v_0, v_2, t]$, and $[v_1, v_2, v_1, t]$.

f) By plugging $\sigma(\rho_0) = 01\,10\,11$, and $\sigma(\rho_f) = 10\,01\,11$, and plugging them into ψ_ϕ, we see that both states satisfy the constraints.

g) We start by expressing this function using quantifiers (like ∀ and ∃), and then unroll these quantifiers into a quantifier-free expression:

$$\begin{aligned}
\psi_{trans}(\mathbf{Z}, \mathbf{Z}') &= \exists i \in \{0,1,2\} : \forall k \in \{0,1,2\} : \begin{cases} \mathbf{z}_k = \mathbf{z}'_k & \text{if } k \neq i \\ \mathbf{z}_k \neq \mathbf{z}'_k & \text{if } k = i \end{cases} \\
&= \big[(\mathbf{z}_0 \neq \mathbf{z}'_0) \cdot (\mathbf{z}_1 = \mathbf{z}'_1) \cdot (\mathbf{z}_2 = \mathbf{z}'_2)\big] + \\
&\quad \big[(\mathbf{z}_0 = \mathbf{z}'_0) \cdot (\mathbf{z}_1 \neq \mathbf{z}'_1) \cdot (\mathbf{z}_2 = \mathbf{z}'_2)\big] + \\
&\quad \big[(\mathbf{z}_0 = \mathbf{z}'_0) \cdot (\mathbf{z}_1 = \mathbf{z}'_1) \cdot (\mathbf{z}_2 \neq \mathbf{z}'_2)\big]
\end{aligned}$$

h) First of all, we will build the state machine with $2^6 = 64$ states. Each state corresponds to a particular routing state ($\mathbf{Z}$). We introduce transitions between two states if and only if they differ in the routing decision of exactly one router. For the final state $\mathbf{Z}_f$, we remove all existing outgoing state transitions, and replace them with a self-looping transition. The initial state of that state machine is the initial state $\sigma(\rho_0)$.

Then, we prepare the the property ϕ', which only holds for states that satisfy both the topological constraints ψ_{topo} and the routing constraints ψ_ϕ. Additionally, the property ϕ_f is defined to be *true* only for the final state $\sigma(\rho_f)$.

Let us express the CTL constraints which hold only for states for which there exists a *safe* sequence of states to migrate to the final state. We wish that there exists a sequence of states, which all satisfy ϕ', and that end in the state ϕ_f.

$$\mathbf{EG}(\phi' \wedge \mathbf{EF}\phi_f)$$

Since the model checker can only find counter-examples, we need to modify the expression above. The goal is that the modified expression will return true only if there exists no such *safe* sequence of states that migrate to the final state. The inverted expression is given as follows:

$$\begin{aligned}
\neg\mathbf{EG}\big(\phi' \wedge \mathbf{EF}\ \phi_f\big) &= \mathbf{AF}\neg\big(\phi' \wedge \mathbf{EF}\ \phi_f\big) \\
&= \mathbf{AF}\big(\neg\phi' \vee \neg\mathbf{EF}\ \phi_f\big) \\
&= \mathbf{AF}\big(\neg\phi' \vee \mathbf{AG}\ \neg\phi_f\big)
\end{aligned}$$

If the model checker will find a counter-example to the inverted expression on the generated state machine, then this counter-example will describe a valid and *safe* sequence of states which can be used to perform the network migration.

i) We need to build one boolean expression that encapsulates all aspects of the migration. Since we know that we can reach the final state in three state transitions, we need to add fours states to the equation: $\mathbf{Z}_0$, $\mathbf{Z}_1$, $\mathbf{Z}_2$, and $\mathbf{Z}_3$. We add the following constraints to the equation: First, we must ensure that the final state $\mathbf{Z}_0$ is the initial state $\sigma(\rho_0)$, and that $\mathbf{Z}_3$ is the final state $\sigma(\rho_f)$. To achieve that, we define the following two characteristic functions:

$$\begin{aligned}
\psi_{\rho_0}(\mathbf{Z}) &= z_0^{\overline{1}} z_0^0\ z_1^1 z_1^{\overline{0}}\ z_2^1 z_2^0 \\
\psi_{\rho_f}(\mathbf{Z}) &= z_0^1 z_0^{\overline{0}}\ z_1^{\overline{1}} z_1^0\ z_2^1 z_2^0
\end{aligned}$$

Second, we must make sure that we can transition from any state to its successor. Finally, every state must both satisfy the topological constraints ψ_{topo}, as well as the constraints from ψ_ϕ. It suffices that we only check $\mathbf{Z}_1$ and $\mathbf{Z}_1$, as we already have verified $\mathbf{Z}_0$ and $\mathbf{Z}_f$ to satisfy our constraints. Hence, we get:

$$\begin{aligned}\psi^* = \psi_0(\mathbf{Z_0}) \ &\cdot \psi_f(\mathbf{Z}_3)\\ &\cdot \psi_{trans}(\mathbf{Z}_0, \mathbf{Z}_1)\\ &\cdot \psi_{trans}(\mathbf{Z}_1, \mathbf{Z}_2)\\ &\cdot \psi_{trans}(\mathbf{Z}_2, \mathbf{Z}_3)\\ &\cdot \psi_{topo}(\mathbf{Z}_1) \cdot \psi_\phi(\mathbf{Z}_1)\\ &\cdot \psi_{topo}(\mathbf{Z}_2) \cdot \psi_\phi(\mathbf{Z}_2)\end{aligned}$$

In fact, there exists only a single solution to the problem above, which is $\mathbf{Z}_0 = \sigma(\rho_0)$, $\mathbf{Z}_1 = 10\,10\,11$, $\mathbf{Z}_2 = 10\,11\,11$, and $\mathbf{Z}_3 = \sigma(\rho_f)$. The resulting ROBDD is drawn in Figure **??**.

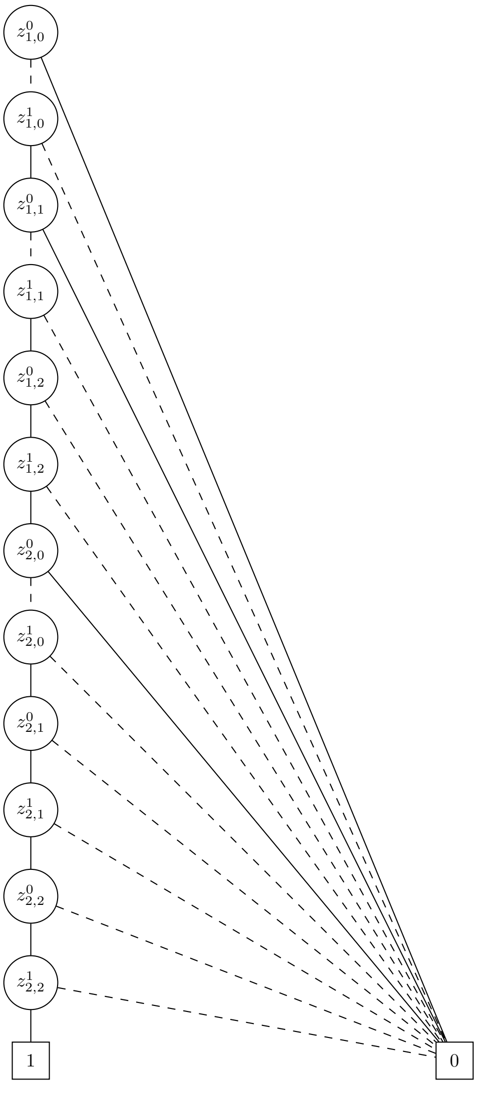

Figure 1: Resulting ROBDD of the complete network migration in 3 steps. This ROBDD only shows the two intermediate states $\mathbf{Z}_1$ and $\mathbf{Z}_2$, since the initial and final states are given.