Prof. Dr. Andreas Podelski
Elisabeth Henkel
Dominik Klumpp

Hand in until January 10th, 2024
15:59 via ILIAS
Discussion: January 15th/16th, 2024

Tutorial for Cyber-Physical Systems - Discrete Models
Exercise Sheet 10

This exercise sheet has one regular task concerning invariants. The remaining tasks are bonus exercises to recap some of the important concepts introduced earlier in the lecture. If you have some time during or after the break, you can use this sheet to practice – and to catch up with the points if needed.

Exercise 1: Invariants — 8 Points

On exercise sheet 8, we used the proposition given below to show that an LT property is not an invariant. In the following task, you will first give a proof for this proposition and then use it to show that the given properties are no invariants.

(a) Let AP be a set of atomic propositions. Prove the following proposition:

Proposition: *Let $E \subseteq (2^{AP})^\omega$ be an LT property. E is not an invariant if and only if there exists a trace $\sigma = A_0 A_1 \ldots \in (2^{AP})^\omega$ such that $\sigma \notin E$, but for every $i \in \mathbb{N}$, the set A_i also occurs in some trace $\pi_i \in E$.*

In your proof, you can assume that for any set $M = \{A_0, A_1, \ldots, A_n\}$ of sets of atomic propositions, there exists a propositional formula Φ such that $A \in M$ if and only if $A \models \Phi$, for all sets $A \in 2^{AP}$.

(b) Let $AP = \{a, b\}$. Consider the following properties:

$$E_1 = \{\, A_0 A_1 \ldots \in (2^{AP})^\omega \mid \forall i \in \mathbb{N} \,.\, (a \in A_i \rightarrow b \in A_{i+1}) \,\}$$
$$E_2 = \{\, A_0 A_1 \ldots \in (2^{AP})^\omega \mid \forall i, j \in \mathbb{N} \,.\, A_i = A_j \,\}$$
$$E_3 = \{\, A_0 A_1 \ldots \in (2^{AP})^\omega \mid |\{i \in \mathbb{N} \mid a \in A_i\}| \geq 2 \,\}$$

For each of them, show that E_k is not an invariant, by giving a trace $\sigma = A_0 A_1 \ldots \notin E_k$, such that every A_i also occurs in some trace π_k that satisfies E_k. For every distinct A_i, also give the trace π_k.

Exercise 2*: LT Properties — 8 Bonus Points

The goal of this task is to learn to identify the different types of LT properties.

Consider the following LT properties with $AP = \{a, b\}$.

(P_1) Always (at any point of time) a or b holds.

(P_2) Either a holds exactly once, or b never holds.

(P_3) If a holds, then b will never hold in the next step.

(P_4) Every time a holds there will be eventually a point of time where b holds.

(P_5) The atomic propositions a and b never hold at the same time.

(P_6) If a holds infinitely often, then b holds infinitely often.

(P_7) There are only finitely many points of time where a holds.

(P_8) True

For each property P_i complete the following tasks:

(a) Formalize P_i using set notation.

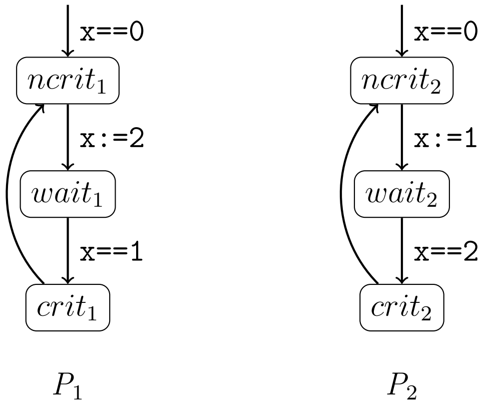
(b) Determine if P_i is an invariant. Explain why or why not.

(c) Determine if P_i is a safety property. Explain why or why not.

(d) Determine if P_i is a liveness property. Explain why or why not.

Exercise 3*: Mutual Exclusion 8 Bonus Points

Consider the following locking protocol. The initial value of the variable `x` is 0.



Note: There is a difference between `x:=1` and `x==1`. The edge labeled with `x:=1` can always be taken (as there is no guard) and it modifies the value of `x`. On the other hand, the edge with `x==1` can only be taken when `x` has the value 1, and it does not modify the value of `x`.

(a) Draw the program graph $P_1 \mid\mid\mid P_2$, i.e. the program graph for the interleaving of P_1 and P_2.

(b) Draw the reachable part of the transition system $\mathcal{T}_{P_1 \mid\mid\mid P_2}$ for the interleaving of the programs. Use the atomic propositions $\{crit_1, crit_2\}$ that are satisfied, whenever process 1 respectively process 2 are in their critical section.

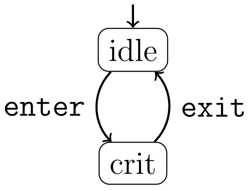
(c) Does the protocol satisfy the mutual exclusion property? Explain your answer in sufficient detail.

(d) Is this a reasonable protocol for parallel programs? Explain your answer in sufficient detail.

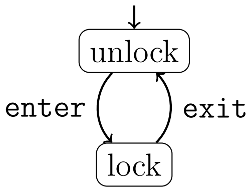
Exercise 4*: Mutual Exclusion without Request — 6 Bonus Points

The goal of this exercise is to help you understand in detail the SOS-rules for parallel compositions.

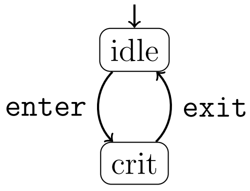
The transition systems below describe a mutual-exclusion protocol with an arbiter. In contrast to the system discussed in the lecture, we omit the *request* action.



TS_1



Arbiter



TS_2

(a) Draw the transition system for the pure interleaving $TS_1 \,|||\, TS_2$. There must be no synchronization between the two transition systems.

For every transition in the interleaving, justify why it must exist using one of the two SOS-rules for pure interleaving.

Example: The interleaving must contain the transition $\langle \mathsf{idle}, \mathsf{idle} \rangle \xrightarrow{\texttt{enter}} \langle \mathsf{crit}, \mathsf{idle} \rangle$ due to the SOS-rule

$$\frac{\mathsf{idle} \xrightarrow{\texttt{enter}}_1 \mathsf{crit}}{\langle \mathsf{idle}, \mathsf{idle} \rangle \xrightarrow{\texttt{enter}} \langle \mathsf{crit}, \mathsf{idle} \rangle}$$

where $\rightarrow_1$ is the transition relation for TS_1. This is an instance of the first of the two SOS-rules,

$$\frac{s_1 \xrightarrow{\alpha}_1 s_1'}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1', s_2 \rangle}$$

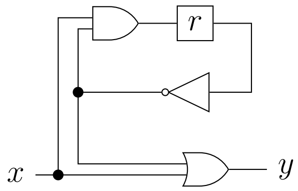
where we set $s_1 = \mathsf{idle}$, $\alpha = \texttt{enter}$, $s_1' = \mathsf{crit}$ and $s_2 = \mathsf{idle}$.

(b) Draw the transition system for the parallel composition $(TS_1 \,|||\, TS_2) \,\|\, \mathit{Arbiter}$ of the transition system from (*a*) with the arbiter. The transition systems must synchronize (*"handshake"*) on the actions $\{\texttt{enter}, \texttt{exit}\}$.

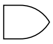
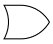
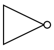
For every transition in the composition, justify why it must exist using one of the three SOS-rules for the synchronization operator.

Exercise 5*: Hardware Circuit — 4 Bonus Points

Consider the following sequential hardware circuit.



Provide the labeled transition system of this hardware circuit (i.e., states are labeled by sets of atomic propositions, transitions are not labeled). The states are the evaluations of the input x and the register r. The transitions represent the stepwise behavior of the circuit. The values of the input x change nondeterministically. The atomic propositions $\{X, Y, R\}$ stand for $x = 1$, $y = 1$ and $r = 1$, respectively. Initially the register r has the value 0 (**false**).

For your reference:  = AND gate,  = OR gate,  = NOT gate