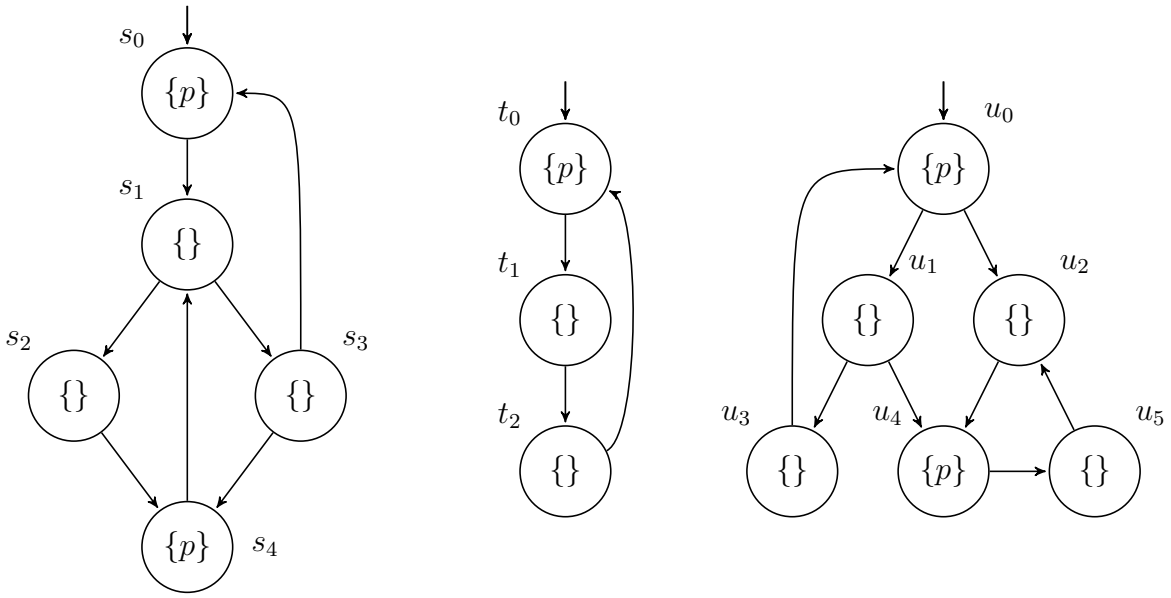
Technische Universität München (I7) **Summer Semester 2019**
J. Esparza / P. Ashok / T. Meggendorfer **11.07.2019**

Model Checking – Exercise sheet 10

Exercise 10.1
Consider the following Kripke structures $\mathcal{K}_1$, $\mathcal{K}_2$, and $\mathcal{K}_3$, over $AP = \{p\}$:



$\mathcal{K}_1$ $\mathcal{K}_2$ $\mathcal{K}_3$

(a) Does $\mathcal{K}_2$ simulate $\mathcal{K}_1$? If yes, give a simulation relation. Otherwise, explain why.

(b) Does $\mathcal{K}_2$ simulate $\mathcal{K}_3$? If yes, give a simulation relation. Otherwise, explain why.

(c) Does $\mathcal{K}_3$ simulate $\mathcal{K}_2$? If yes, give a simulation relation. Otherwise, explain why.

(d) Does $\mathcal{K}_3$ simulate $\mathcal{K}_1$? If yes, give a simulation relation. Otherwise, explain why.

Exercise 10.2
Let $\mathcal{K}_1$, $\mathcal{K}_2$, and $\mathcal{K}_3$ be Kripke structures. Show that if $\mathcal{K}_1$ and $\mathcal{K}_2$ are bisimilar, and $\mathcal{K}_2$ and $\mathcal{K}_3$ are bisimilar, then $\mathcal{K}_1$ and $\mathcal{K}_3$ are also bisimilar.

Exercise 10.3

Consider the following program with a Boolean variable `x`. Initially, the value of `x` is `false`. The question mark stands for a nondeterministic value.

```
1 x = ?;
2 while (x)
3         x = ?;
4 while (true) {}
```

Let $AP = \{x\}$, where x is true only in states where the variable `x` is `true`.

(a) Construct a Kripke structure $\mathcal{K} = (S, \rightarrow, r, AP, \nu)$ for the above program.

(b) Let $\approx$ be an equivalence relation on S such that for all $s \approx t$ we have $\nu(s) = \nu(t)$. Construct from $\mathcal{K}$ the abstracted Kripke structure $\mathcal{K}'$ w.r.t. $\approx$.

(c) Model check the following formulas with $\mathcal{K}'$. Refine the abstraction if necessary.

(i) $\neg x \, \mathbf{W} \, x$

(ii) $\mathbf{G}(\neg x \rightarrow \mathbf{X} \neg x)$

(iii) $\mathbf{X}(\neg x \rightarrow \mathbf{G} \neg x)$

Exercise 10.4

We consider the following program, over the integer variables x and y:

```
1 if (x >= 0) x = -x;
2 if (y >= 0) y = -y;
3 if (x + y > 0) error;
4 end
```

1. Give the set of configurations of the program (some may not be reachable).
2. Draw the abstract transition system with the predicates l_1, l_2, l_3, l_4 and "error".
3. Give a path ρ in the abstract transition system reaching a state where "error" holds.
4. What is the longest prefix (denoted ρ') of ρ that can be concretized ?
5. Denote q the state in the abstract transition system reached by ρ'. Give a predicate that separates configurations reachable by ρ' from configurations that admit a successor.
6. Draw the abstract transition system with that additional predicate.
7. How many times does we have to repeat the abstraction refinement technique to exhibit an abstract transition system that does not reach the error state ? Draw that transition system, how many predicates have we introduced ?

Solution 10.1

(a) Yes. $H = \{(s_0, t_0), (s_1, t_1), (s_2, t_2), (s_3, t_2), (s_4, t_0)\}$.

(b) No. If there exists a simulation H from $\mathcal{K}_3$ to $\mathcal{K}_2$, then we know that $(u_0, t_0) \in H$. Since $u_0 \to u_1$, we have $(u_1, t_1) \in H$. However, $u_1 \to u_4$ and u_4 satisfies p, but no successors of t_1 satisfy p, so H cannot exist.

(c) Yes. $H = \{(t_0, u_0), (t_1, u_1), (t_2, u_3\}$.

(d) Yes. $H = \{(s_0, u_0), (s_1, u_1), (s_2, u_3), (s_3, u_3), (s_4, u_0)\}$. Alternatively, we can also prove that $\mathcal{K}_1$ and $\mathcal{K}_2$ are bisimilar and use the result from (c).

Solution 10.2

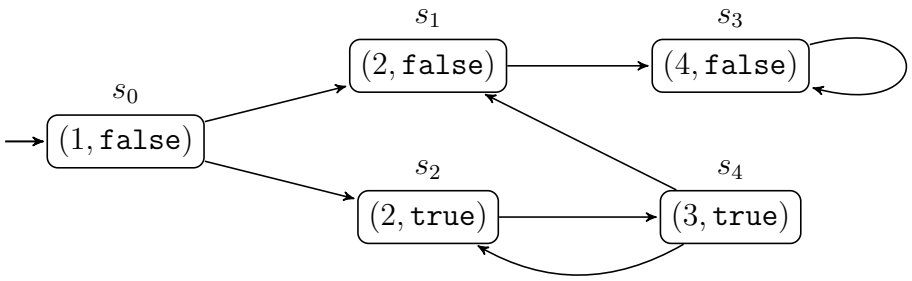
Let H_{12} be a bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$ and H_{23} be a bisimulation between $\mathcal{K}_2$ and $\mathcal{K}_3$. We define $H_{13} = \{(s, u) \mid \exists t : (s, t) \in H_{12} \wedge (t, u) \in H_{23}\}$ and show that H_{13} is a bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_3$.

First, we prove that H_{13} is a simulation from $\mathcal{K}_1$ to $\mathcal{K}_3$. Basically, we need to prove that if $(s, u) \in H_{13}$ and $s \to_1 s'$, then there exists u' such that $u \to_3 u'$ and $(s', u') \in H_{13}$. From the definition of $(s, u) \in H_{13}$, we know that there exists t such that $(s, t) \in H_{12}$ and $(t, u) \in H_{23}$. Since $(s, t) \in H_{12}$ and $s \to_1 s'$, there must exist t' such that $t \to_2 t'$ and $(s', t') \in H_{12}$. Similarly, since $(t, u) \in H_{23}$ and $t \to_2 t'$, there must exist u' such that $u \to_3 u'$ and $(t', u') \in H_{23}$. Because $(s', t') \in H_{12}$ and $(t', u') \in H_{23}$, by the definition of H_{13} we have $(s', u') \in H_{13}$.

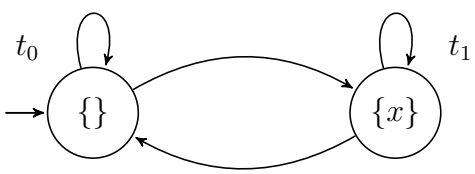
Analogously, we can prove that $\{(u, s) \mid (s, u) \in H_{13}\}$ is a simulation from $\mathcal{K}_3$ to $\mathcal{K}_1$.

Solution 10.3

(a) Each state of the following Kripke structure $\mathcal{K}$ is a pair of a program location and a valuation of `x`.



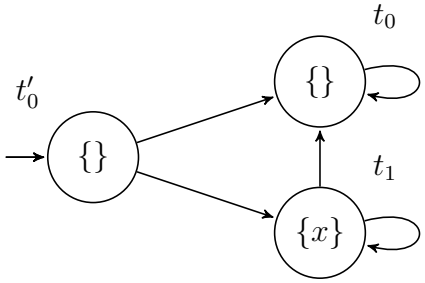
(b) Let $t_0 = [s0] = \{s_0, s_1, s_3\}$ and $t_1 = [s1] = \{s_2, s_4\}$. The abstraction $\mathcal{K}'$ is as follows:



(c) (i) $\mathcal{K}' \models \neg x \; \mathbf{W} \; x$

(ii) $\mathcal{K}' \not\models \mathbf{G}(\neg x \rightarrow \mathbf{X}\neg x)$. A counterexample in $\mathcal{K}'$ is $t_0 t_1 t_1^\omega$, which corresponds to the run $s_0 s_2 (s_4 s_2)^\omega$ in $\mathcal{K}$. So, $\mathcal{K} \not\models \mathbf{G}(\neg x \rightarrow \mathbf{X}\neg x)$.

(iii) $\mathcal{K}' \not\models \mathbf{X}(\neg x \rightarrow \mathbf{G}\neg x)$. A counterexample in $\mathcal{K}'$ is $t_0 t_0 t_1^\omega$. However, there are no corresponding runs in $\mathcal{K}$ because such paths must start with $s_0 s_1$, but no successors of s_1 are in t_1. Since $s_0 \in t_0$ and s_0 has a successor in t_1, we can refine the abstraction to distinguish s_0 from s_1. $t'_0 = \{s_0\}$ and $t_0 = \{s_1, s_3\}$, and construct a new Kripke structure $\mathcal{K}''$ as follows.



We have $\mathcal{K}'' \models \mathbf{X}(\neg x \rightarrow \mathbf{G}\neg x)$.

Solution 10.4

1. $\{l_1, l_2, l_3, l_4, error\} \times \mathbb{N} \times \mathbb{N}$

2. The states of the abstract TS are $\{l_1, l_2, l_3, l_4, error\}$ and the transition relation is $\{(l_1, l_2), (l_2, l_3), (l_3, error), (l_3, l_4)\}$.

3. $\rho = l_1 \; l_2 \; l_3 \; error$

4. $\rho' = l_1 \; l_2 \; l_3$

5. $q = l_3$. Configurations reachable by ρ' will satisfy the predicate $(x \leq 0) \wedge (y \leq 0)$ because if either of x or y are positive, l_1 and l_2 will make them non-positive. However, all configurations in l_3 admit a successor.

6. New states will be $\{l_1, l_2, l_3, l_4, error\} \times \{p_1, \neg p_1\}$ where p_1 is the predicate $(x \leq 0) \wedge (y \leq 0)$. The transition relation would be
$\{((l_1, p_1), (l_2, p_1)), ((l_2, p_1), (l_3, p_1)), ((l_3, p_1), (l_4, p_1))\} \cup$
$\{((l_1, \neg p_1), (l_2, \neg p_1)), ((l_1, \neg p_1), (l_2, p_1))\} \cup$
$\{((l_2, \neg p_1), (l_3, \neg p_1)), ((l_2, \neg p_1), (l_3, p_1))\} \cup$
$\{((l_3, \neg p_1), (error, \neg p_1)), ((l_3, \neg p_1), (l_4, \neg p_1))\}$.

7. From the above refinement, we realize that there is a state $(l_3, \neg p_1)$, which admits a successor to an *error* state; however there is no concrete path which leads to $(l_3, \neg p_1)$. So we try to find a predicate which separates the configuration that can be reached by

the path $(l_1, \neg p_1)(l_2, \neg p_1)$ from the configurations in $(l_2, \neg p_1)$ which have a successor to $(l_3, \neg p_1)$...