Prof. Dr. Andreas Podelski
Matthias Heizmann

25.1.2012
Submission: 31.1.2012
at the beginning of the lecture

Tutorials for Program Verification
Exercise sheet 12

Exercise 1: Transition Invariants 2+4 points

Let R be a transition relation. In the lecture a transition invariant T was defined *inductive* if $T \circ R \subseteq T$. We can adapt the definition of *inductivity* to a set of abstract transitions $\{T_1, \dots, T_n\}$ in the following two ways.

Definition 1 We call $\{T_1, \dots, T_n\}$ *inductive* if for all i there exists j such that $T_i \circ R \subseteq T_j$.

Definition 2 We call $\{T_1, \dots, T_n\}$ *inductive* if $(T_1 \cup \dots \cup T_n) \circ R \subseteq T_1 \cup \dots \cup T_n$.

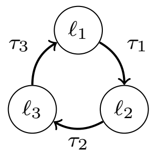
(a) Are both definitions equivalent? If not give a counterexample.

(b) For which of the two defintions above is the set of abstract transitions $P^{\#}$ computed by the TPA algorithm inductive? Prove your claims.

Exercise 2: Termination and Non-Termination 5 points

(a) Consider the following program $P = (\Sigma, \mathcal{T}, \rho)$, where every state is an inital state.

```
while (x >= 0) {
   x:=x-y;
   y:=y+1;
}
```



$$\begin{array}{rl}
\Sigma \text{ is} & \{\ell_1, \ell_2, \ell_3\} \times \mathbb{Z} \times \mathbb{Z} \\
\rho_{\tau_1} \text{ is} & pc = \ell_1 \wedge pc' = \ell_2 \wedge x' = x \wedge y' = y \wedge x \geq 0 \\
\rho_{\tau_2} \text{ is} & pc = \ell_2 \wedge pc' = \ell_3 \wedge x' = x - y \wedge y' = y \\
\rho_{\tau_3} \text{ is} & pc = \ell_3 \wedge pc' = \ell_1 \wedge x' = x \wedge y' = y + 1
\end{array}$$

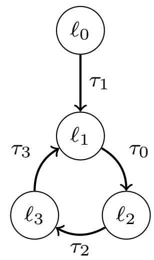
Is the program terminating? If the program terminates give either

- a disjunctively well-founded transition relation
- or a ranking-function whose value is decreased after the execution of every single transition.

If the program does not terminate describe some infinite program execution.

(b) Consider the following program $P = (\Sigma, \Sigma_{\mathsf{init}}\mathcal{T}, \rho)$, where Σ_{init} denotes the set of initial states.

```
0: if (y!=0) {
1:     while (-42<x && x>42 && z<0) {
2:         x := x+y;
3:         y := y*z;
       }
   }
```



$$
\begin{array}{rl}
\Sigma \text{ is} & \{\ell_1, \ell_2, \ell_3\} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \\
\Sigma_{\mathsf{init}} \text{ is} & pc = \ell_0 \\
\rho_{\tau_0} \text{ is} & pc = \ell_0 \wedge pc' = \ell_1 \wedge x' = x \wedge y' = y \wedge z' = z \wedge y \neq 0 \\
\rho_{\tau_1} \text{ is} & pc = \ell_1 \wedge pc' = \ell_2 \wedge x' = x \wedge y' = y \wedge z' = z \wedge -42 < x \wedge x > 42 \wedge z < 0 \\
\rho_{\tau_2} \text{ is} & pc = \ell_2 \wedge pc' = \ell_3 \wedge y' = y \wedge z' = z \wedge x' = x + y \\
\rho_{\tau_3} \text{ is} & pc = \ell_3 \wedge pc' = \ell_1 \wedge x' = x \wedge z' = z \wedge y' = y \cdot z
\end{array}
$$

Do all program executions that start in an initial state terminate? If your answer ist *yes* give an explanation, if your answer is *no* give a recurrence set for the while loop.