6.0/4.0 VU Formale Methoden der Informatik

185.291 SS 2013 06 December 2013

Kennzahl (study id)	Matrikelnummer (student id)	Familienname (family name)	Vorname (first name)	Gruppe (version)
				A

1.) Consider the following problem:

> **SOLVE-EQUATION**
>
> INSTANCE: Two programs Π_1 and Π_2 which take an arbitrary integer (i.e., positive, 0, or negative) as input and return an integer value. It is guaranteed that Π_1 and Π_2 terminate on any input.
>
> QUESTION: Does there exist an integer v such that on input v the programs Π_1 and Π_2 return the same value, i.e. $\Pi_1(v) = \Pi_2(v)$?

Prove that the problem **SOLVE-EQUATION** is semi-decidable. For this, describe a procedure that shows the semi-decidability of the problem (i.e. a semi-decision procedure for **SOLVE-EQUATION**) and argue that it is correct.

(15 points)

2.) (a) Use Ackermann's reduction and translate

$$A(A(x)) \doteq A(B(x)) \rightarrow B(A(B(x))) \doteq y \vee C(x, y) \doteq C(A(x), B(x))$$

to a validity-equivalent E-formula φ^E. A, B, and C are function symbols, x and y are variables. **(4 points)**

(b) Show: φ is satisfiable iff $\neg\varphi$ is *not* valid. **(3 points)**

(c) Let φ^{uf} be an equality formula containing uninterpreted functions. Let $FC^E(\varphi^{uf})$ and $flat^E(\varphi^{uf})$ be obtained by Ackermann's reduction. Prove the following.

$$\varphi^{uf} \text{ is satisfiable} \quad \text{iff} \quad FC^E(\varphi^{uf}) \wedge flat^E(\varphi^{uf}) \text{ is satisfiable.}$$

Hints:
H1: φ^{uf} is valid iff $FC^E(\varphi^{uf}) \rightarrow flat^E(\varphi^{uf})$ is valid.
H2: $flat^E(\neg\varphi^{uf}) = \neg flat^E(\varphi^{uf})$.
H3: $FC^E(\varphi^{uf}) = FC^E(\neg\varphi^{uf})$.

(8 points)

3.) Show that the following correctness assertion is totally correct. Describe the function computed by the program; assume x and y to be the inputs and z the output of the program.

$\{\, Pre\colon x \geq 1 \wedge y \geq 2 \,\}$
$u := y;$
$z := 0;$
while $u \leq x$ do
    $u := u * y;$
    $z := z + 1$
od
$\{\, Post\colon y^z \leq x < y^{z+1} \,\}$

Hints: Use the invariant $Inv\colon u = y^{z+1} \wedge y \leq u \leq xy \wedge y \geq 2$. For determining the function computed by the program, it is not necessary to evaluate the program; just analyse the postcondition. Depending on how you choose the variant, use one of the following annotation rules:

while e do $\cdots$ od $\mapsto$ $\{\, Inv \,\}$while e do $\{\, Inv \wedge e \wedge t{=}t_0 \,\} \cdots \{\, Inv \wedge 0{\leq}t{<}t_0 \,\}od\{\, Inv \wedge \neg e \,\}$
while e do $\cdots$ od $\mapsto$ $\{\, Inv \,\}$while e do $\{\, Inv \wedge e \wedge t{=}t_0 \,\} \cdots \{\, Inv \wedge (e \Rightarrow 0{\leq}t{<}t_0) \,\}od\{\, Inv \wedge \neg e \,\}$

(15 points)

4.) Bisimulation.

Let $M_1 = (S_1, I_1, R_1, L_1)$ and $M_2 = (S_2, I_2, R_2, L_2)$ be two Kripke structures.

Simulation

Remember, a relation $H \subseteq S_1 \times S_2$ is a simulation relation if for each $(s, s') \in H$ holds:

- $L_1(s) = L_2(s')$, and
- for each $(s, t) \in R_1$ there is a $(s', t') \in R_2$ such that $(t, t') \in H$.

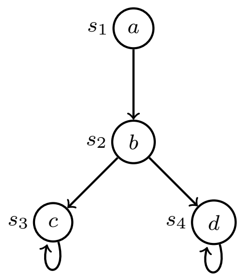
Further remember, M_2 *simulates* M_1, in signs $M_1 \leq M_2$, if there is a simulation relation $H \subseteq S_1 \times S_2$ *such that*

- for each initial state $s \in I_1$ there is an initial state $s' \in I_2$ with $(s, s') \in H$.

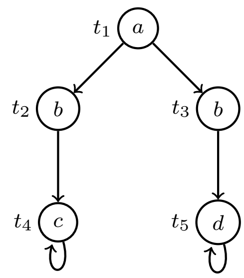
In the following, we say that H *witnesses the similarity of* M_1 *and* M_2 in case H is a simulation relation from M_1 to M_2 that satisfies the condition stated above.

(a) Show that there is no simulation relation H that witnesses $M_1 \leq M_2$.

(M_1)



(M_2)



(3 points)

(b) Show that both M_1 and M_2 from task (a) satisfy the same LTL formulae, i.e., for every LTL formula ϕ it holds:

$$M_1 \models \phi \text{ if and only if } M_2 \models \phi$$

(3 points)

(c) Show that both M_1 and M_2 from task (a) do *not* satisfy the same CTL formulae, i.e., there is a CTL formula ϕ such that:

$$M_1 \models \phi \text{ and } M_2 \not\models \phi$$

(3 points)

(d) **CTL Model Checking Algorithm**

Let $K = (S, T, L)$ be a Kripke structure and let p, q be atomic propositions. Give an algorithm that computes the set of all states $s \in S$ that satisfy $\mathbf{A}[p\mathbf{U}q]$.

(6 points)