1	2	3	4	Σ	Grade

6.0/4.0 VU Formale Methoden der Informatik

185.291 May, 21 2021

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

1.) (a) Recall the **HALTING** problem which takes a program and a string as input, and consider the following variant thereof:

> **HALTING-X**
>
> INSTANCE: Two program Π_1, Π_2 that take a string as input.
>
> QUESTION: Is it true, that for all input strings I: if Π_1 halts on I then Π_2 halts on I.

The following function f provides a polynomial-time many-one reduction from **HALTING** to **HALTING-X**: for a program Π and a string I let $f(\Pi, I) = (\Pi_1, \Pi_2)$ with

$$\begin{aligned} \Pi_1(\texttt{string } S) &= \texttt{return;} \\ \Pi_2(\texttt{string } S) &= \texttt{if } (S\ = I)\ \{\texttt{call } \Pi(S);\}\ \texttt{return;} \end{aligned}$$

Show that (Π, I) is a yes-instance of **HALTING** $\Longleftrightarrow$ (Π_1, Π_2) is a yes-instance of **HALTING-X**.

(9 points)

(b) Tick the correct statements (for ticking a correct statement a certain number of points is given; ticking an incorrect statement results in a substraction of the same amount; you cannot go below 0 points):

○ Since **HALTING** is undecidable, our reduction from (a) shows that **HALTING-X** is undecidable.

○ Since **HALTING** is semi-decidable, our reduction from (a) shows that **HALTING-X** is semi-decidable.

○ Since **HALTING** is undecidable, our reduction from (a) shows that there is no SIMPLE program that solves **HALTING-X**.

○ Since **HALTING** is semi-decidable, our reduction from (a) shows that there is a SIMPLE program that solves **HALTING-X**.

○ If we would have a decision procedure for **HALTING-X**, we can solve **HALTING** using our reduction from (a).

○ If we would have a decision procedure for **HALTING**, we can solve **HALTING-X** using our reduction from (a).

(6 points)

2.) (a) Consider the theory $\mathcal{T}_A$ of arrays and the following formula

$$\varphi: \quad a\langle \ell \triangleleft v\rangle[k] \doteq b[k] \ \wedge\ b[k] \ \not\doteq v \wedge a[k] \ \doteq v \wedge \big(\forall i \ (i \not\doteq \ell \rightarrow a[i] \doteq b[i])\big) \ .$$

If φ is $\mathcal{T}_A$-sat, then provide a $\mathcal{T}_A$-model for φ. For the proposed model, you have to show that it satisfies all axioms of $\mathcal{T}_A$ and φ.

If φ is $\mathcal{T}_A$-unsat, then provide a proof in the semantic argument method (similarly to the proofs in the lecture and on the extra sheets). If you use a derived rule, you have to prove the correctness of the rule in the same method.

Besides the equality axioms reflexivity, symmetry and transitivity, you have the following ones for arrays.

- $\forall a, i, j \ \big(i \doteq j \ \rightarrow \ a[i] \doteq a[j]\big)$ (array congruence)
- $\forall a, v, i, j \ \big(i \doteq j \ \rightarrow \ a\langle i \triangleleft v\rangle[j] \doteq v\big)$ (read-over-write 1)
- $\forall a, v, i, j \ \big(i \not\doteq j \ \rightarrow \ a\langle i \triangleleft v\rangle[j] \doteq a[j]\big)$ (read-over-write 2)

(10 points)

(b) Apply the sparse method to the following E-formula

$$\psi^E: a \doteq b \wedge (b \not\doteq d \rightarrow (b \not\doteq c \rightarrow c \not\doteq d)) \wedge (d \not\doteq e \wedge e \not\doteq c \rightarrow c \not\doteq d)$$

and derive a short satisfiability-equivalent propositional formula consisting of the propositional skeleton and the transitivity constraints. Name each step in the sparse method and explain briefly, why you apply the step or why you don't. **(5 points)**

3.) (a) Let p be the following IMP program:

```
$x := 0; y := 0;$
**while** $x < n$ **do**
   $y := y + 3 * x;$
   $x := x + 1$
**od**
```

Give a loop invariant and variant for the **while** loop in p and prove the validity of the total correctness triple $[n = 16]\ p\ [y = 360]$.

(10 points)

(b) Provide a non-trivial pre-condition A and a non-trivial post-condition B, such that the total correctness triple

$[A]\ x := 1;\ \textbf{abort}\ [B]$ is valid.

Trivial means equivalent to `true` or `false`, so your precondition A and postcondition B should not be equivalent to `true` or `false`. In case such a A and/or B does not exist, explain why there exist no such A and/or B.

(3 points)

(c) Consider the partial correctness triple

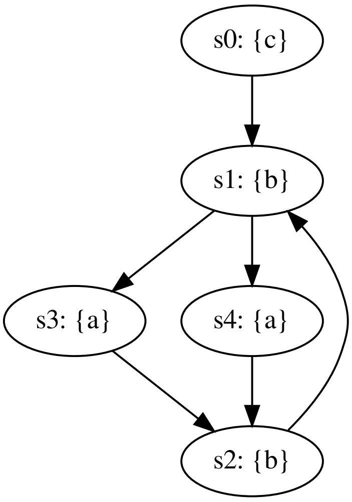
$$\{x \geq 0\}\ y = 5 * x\ \{y > x\}$$

Is this triple valid? If so, give a formal proof. Otherwise, give a counterexample.

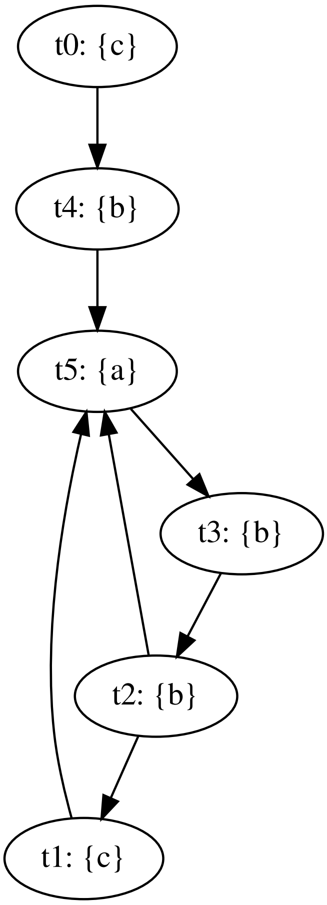
(2 points)

4.) (a) Provide a non-empty simulation relation H that witnesses $M_1 \leq M_2$, where M_1 and M_2 are shown below. The initial state of M_1 is s_0, the initial state of M_2 is t_0:

Kripke structure M_1:

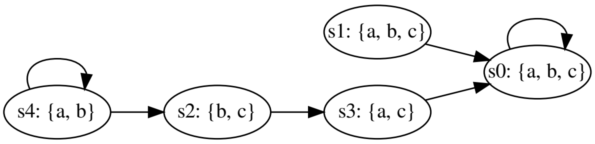


Kripke structure M_2:



(4 points)

(b) Consider the following Kripke structure M:



For each of the following formulae φ,

i. indicate whether the formula is in CTL, LTL, and/or CTL*, and

ii. list the states s_i on which the formula φ holds; i.e. for which states s_i do we have $M, s_i \models \varphi$?
(If φ is a path formula, list the states s_i such that $M, s_i \models \mathbf{A}\varphi$.)

φ	CTL	LTL	CTL*	States s_i
$\mathbf{G}(a)$	□	□	□	
$\mathbf{AX}(c)$	□	□	□	
$\mathbf{E}[(c)\ \mathbf{U}\ (a)]$	□	□	□	
$\mathbf{AG}(a \wedge b)$	□	□	□	
$\mathbf{F}(c)$	□	□	□	

(5 points)

(c) **LTL tautologies**

Prove that the following formulas are tautologies, i.e., they hold for every Kripke structure M and every path π in M, or find a Kripke structure M and path π in M, for which the formula does not hold and justify your answer.

i. $(\mathbf{G}a \Rightarrow \mathbf{GF}b) \Rightarrow \mathbf{G}(\mathbf{G}a \Rightarrow \mathbf{F}b)$

ii. $\mathbf{G}(\mathbf{G}a \Rightarrow \mathbf{F}b) \Rightarrow (\mathbf{G}a \Rightarrow \mathbf{GF}b)$

(6 points)