1	2	3	4	Σ

6.0/4.0 VU Formale Methoden der Informatik (185.291)

Apr 16, 2021

Kennz. (study id)	Matrikelnummer (student id)	Nachname (surname)	Vorname (first name)

1.) Consider the following variant of the dominating set problem **DOM**:

DOMINATING SET VARIATION (DOM)

INSTANCE: A directed graph $G = (V, E)$ and a subset $X \subseteq V$ of forbidden vertices.

QUESTION: Does there exist a set $S \subseteq V$ such that $S \cap X = \emptyset$ and for each $v \in V$ either $v \in S$ or there is an $u \in S$ with $(u, v) \in E$.

(a) The following function f provides a polynomial-time many-one reduction from **3SAT** to **DOM**: for a 3-CNF formula $\varphi = \bigwedge_{j=1}^{m}(l_{j1} \vee l_{j2} \vee l_{j3})$ over atoms $A = \{a_1, \dots, a_n\}$ let $f(\varphi) = (G, X)$, where $G = (V, E)$ with

$$
\begin{aligned}
V &= \{v_1, v_1', \dots, v_n, v_n', c_1, \dots, c_m\}; \\
E &= \{(v_i, v_i'), (v_i', v_i) \mid 1 \leq i \leq n\} \cup \\
&\quad\ \{(v_i, c_j) \mid a_i \in \{l_{j1}, l_{j2}, l_{j3}\}, 1 \leq i \leq n, 1 \leq j \leq m\} \cup \\
&\quad\ \{(v_i', c_j) \mid \neg a_i \in \{l_{j1}, l_{j2}, l_{j3}\}, 1 \leq i \leq n, 1 \leq j \leq m\}; \text{ and} \\
X &= \{c_1, \dots, c_m\}.
\end{aligned}
$$

It holds that φ is a yes-instance of **3SAT** $\iff$ $f(\varphi)$ is a yes-instance of **DOM**.

Show the $\Longrightarrow$ direction of the statement.

(9 points)

(b) In what follows assume the reduction from **3SAT** to **DOM** is correct, and further assume we have shown that **DOM** is in NP. Also recall that **3SAT** is NP-complete.

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):

- **DOM** is NP-complete
- **DOM** is NP-hard
- there exists a polynomial-time many-one reduction from **DOM** to **SAT**
- **DOM** is decidable
- a polynomial-time many-one reduction from **DOM** to **SAT** shows P=NP
- **DOM** is in P

(6 points)

2.) (a) Consider the implementation of the function `add4` in `C`, which is supposed to compute $4x$ for an unsigned 32 bit integer x.

```c
uint32_t add4(uint32_t x){
  uint32_t y;

  y = x + x + x + x;
  return y;
}
```

Suppose the function is called with a parameter of correct type. Does this function return the mathematically correct value $4x$? If your answer is yes, then *prove the correctness* of the function. Otherwise describe *exactly and in detail* what is going on. Would you answer differently, if `y` is of type `unit64_t`? Explain.

(4 points)

(b) Consider the following clause set $\hat{\delta}(\varphi)$ which has been derived from an (unknown) formula φ by Tseitin's translation (atoms have not been labeled).

C_1:	$\ell_1 \vee \neg x \vee \neg y$	C_2:	$\neg \ell_1 \vee x$	C_3:	$\neg \ell_1 \vee y$
C_4:	$\neg \ell_2 \vee \neg y \vee z$	C_5:	$\ell_2 \vee y$	C_6:	$\ell_2 \vee \neg z$
C_7:	$\neg \ell_3 \vee \neg \ell_1 \vee z$	C_8:	$\ell_3 \vee \ell_1$	C_9:	$\ell_3 \vee \neg z$
C_{10}:	$\neg \ell_4 \vee \neg x \vee \ell_2$	C_{11}:	$\ell_4 \vee x$	C_{12}:	$\ell_4 \vee \neg \ell_2$
C_{13}:	$\neg \ell_5 \vee \neg \ell_4 \vee \ell_3$	C_{14}:	$\ell_5 \vee \ell_4$	C_{15}:	$\ell_5 \vee \neg \ell_3$

(i) Reconstruct φ from $\hat{\delta}(\varphi)$ using the smallest number of connectives.

(ii) Start from $\hat{\delta}(\varphi)$ and extend it by a single nonempty clause C in such a way that φ is valid iff $\hat{\delta}(\varphi) \wedge C$ is unsatisfiable.

(iii) Use resolution to prove the validity of φ (no additional translation is allowed!).

(5 points)

(c) Let R be $\forall x\, p(x,x)$, and let φ be $\exists x \exists y \forall z \big[p(x,y) \wedge p(y,z)\big]$, where p is a binary predicate symbol. Check whether $\varphi \models R$ holds. If yes, then give a proof; otherwise give a counter-example and prove that the entailment does not hold. **(6 points)**

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

$$
\begin{array}{l}
x := 0; y := 0; z := 1; \\
\textbf{while } z < n \textbf{ do} \\
\quad x := x + 3; \\
\quad y := y + 4 * x; \\
\quad z := z + 1 \\
\textbf{od}
\end{array}
$$

Give a loop invariant for the **while** loop in p and prove the validity of the partial correctness triple $\{n > 1\}\ p\ \{y = 2 * x * n\}$.

Hint: Make sure that your invariant expresses equalities among y, z, x, as well as equalilties among z, x.

(9 points)

(b) Let p be an IMP program such that $\{true\}\ p\ \{x = -2 \wedge y = 2\}$ is valid.
Is $\{y = 2\}\ p\ \{y \geq 0\}$ valid? If so, give a formal proof. Otherwise, give a counterexample.

(3 points)

(c) Let p be the IMP program

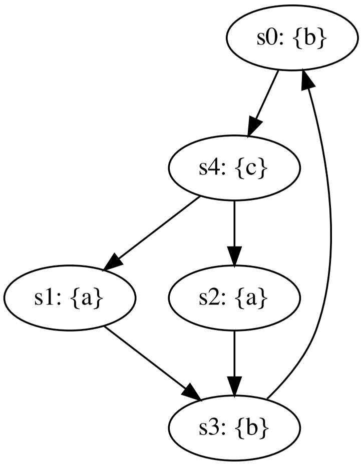
$$\textbf{while } x > 0 \textbf{ do } x := x - 2$$

Give a pre-condition A such that $[A]\ p\ [x = -1]$ is valid. Your precondition A should not be $x = -1$ and it should not be equivalent to `true` nor to `false`.

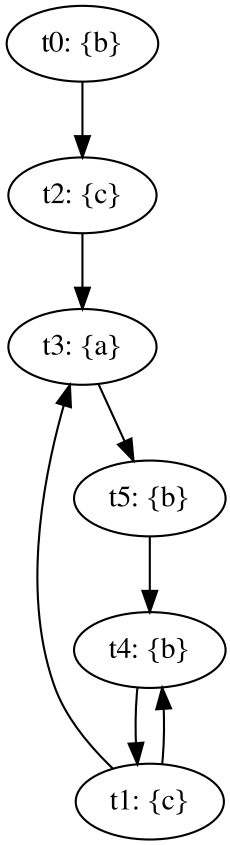
(3 points)

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:

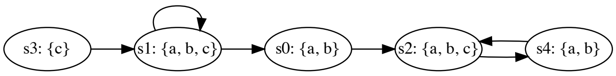


Kripke structure M_2:



(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
$\mathbf{X}(c)$	☐	☐	☐	
$\mathbf{AG}(a \wedge b)$	☐	☐	☐	
$\mathbf{G}(a)$	☐	☐	☐	
$\mathbf{E}[c\ \mathbf{U}\ c]$	☐	☐	☐	

(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. $((\mathbf{G}b)\ \mathbf{U}\ (\mathbf{X}a)) \Rightarrow \mathbf{F}(a \wedge \mathbf{G}b)$

ii. $(\mathbf{F}(a \wedge \mathbf{G}b)) \Rightarrow ((\mathbf{G}b)\ \mathbf{U}\ (\mathbf{X}a))$

(6 points)

Grading scheme: 0–29 nicht genügend, 30–35 genügend, 36–41 befriedigend, 42–47 gut, 48–60 sehr gut