6.0/4.0 VU Formale Methoden der Informatik (185.291)

July 1, 2016

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

1.) Consider the following problem:

> **SAT-UNSAT**
>
> INSTANCE: A pair (φ_1, φ_2), where φ_1 and φ_2 are propositional formulas.
>
> QUESTION: Is it the case that φ_1 is satisfiable or φ_2 is unsatisfiable?

By providing a suitable reduction from an NP-complete problem, prove that **SAT-UNSAT** is an NP-hard problem. Argue formally that your reduction is correct. **(15 points)**

2.) (a) Consider the C program shown below. Justify your answers to the questions in detail.

```
int main(void) {
  unsigned int x = 10;
  unsigned int y = (1 << 16);
  while (x >= 0 && y >= 0) {
      y = y * y;
      x = x - y;
    }
}
```

- Does the program terminate?
- Does termination depend on the integer representation?

(6 points)

(b) Let ϕ be a propositional formula in CNF. For a literal ℓ, we define

$$S(\phi, \ell) := \{C \mid C \in \phi \text{ and } \ell \in C\}$$

to be the set of all clauses in ϕ which contain the literal ℓ.

Let $C \in \phi$ be a clause. Assume that there exists a literal $\ell \in C$ such that for *every* resolvent R of C with a clause $C' \in S(\phi, \neg\ell)$ (i.e. R is the result of resolving upon the variable of ℓ) it holds that R contains both a literal x and a literal $\neg x$ of some variable x in ϕ. Consider the formula $\phi' := \phi \setminus \{C\}$ obtained from ϕ by removing the clause C. Prove that ϕ is satisfiable if and only if ϕ' is satisfiable. **(9 points)**

3.) Show that the following correctness assertion is totally correct. Describe the function computed by the program if we consider x as its input and y as its output.

Hints: Use the formula $i = (y+1)^3 \wedge 0 \le y^3 \le x$ as loop invariant. Depending on how you choose the variant, use one of the following annotation rules:

$\mathsf{while}\ e\ \mathsf{do} \cdots \mathsf{od} \mapsto \{\,Inv\,\}\mathsf{while}\ e\ \mathsf{do}\ \{\,Inv{\wedge}e{\wedge}t{=}t_0\,\} \cdots \{\,Inv \wedge 0{\le}t{<}t_0\,\}\mathsf{od}\{\,Inv{\wedge}\neg e\,\}$

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

$\{\,x \ge 0\,\}$
$i := 1;$
$y := 0;$
while $i \le x$ do
    $j := 6 + y * 3;$
    $y := 1 + y;$
    $i := 1 + y * j + i$
od
$\{\,y^3 \le x < (y+1)^3\,\}$

(15 points)

4.) Model Checking

(a) In the lecture we saw that we can use CBMC to solve NP-complete decision problems. In this exercise, you are to illustrate this approach for the *dominating set problem.* All the graphs in this question are simple and undirected.

A *dominating set* of a graph $G = (V, E)$ is a set $D \subseteq V$ such that every vertex not in D is adjacent to a vertex in D. In other words, for all vertices $v \in V$, either $v \in D$ or there exists a vertex $w \in D$ such that the edge $\{v, w\}$ belongs to E.

In the *dominating set problem*, you are given a graph $G = (V, E)$ and a number K and have to decide whether there is a dominating set D for G that contains at most K vertices.

Write a C program that implements a *guess and check* routine for the dominating set problem and instrument the program with an appropriate CBMC assertion.

You may assume the following template:

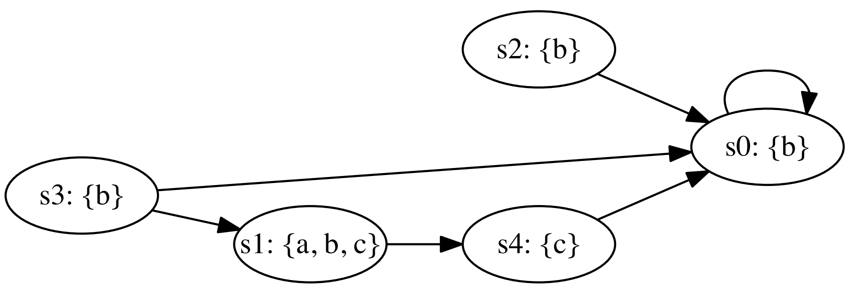
```
int nondet_bool(); // non-deterministically returns 0 or 1

// Fixed sample input:
#define N 6  // number of vertices in the graph
#define K 3  // max. number of vertices in the dominating set
// Adjacency matrix of the graph, i.e. edge[i][j] = 1 iff {i,j} in E
int edge[N][N] = { {0, 1, 0, 0, 0, 0},
                   {1, 0, 1, 1, 0, 0},
                   {0, 1, 0, 1, 1, 0},
                   {0, 1, 1, 0, 0, 1},
                   {0, 0, 1, 0, 0, 0},
                   {0, 0, 0, 1, 0, 0} };

int main() {
  // add code here:
  // 1. guess a solution
  // 2. put an assertion such that CBMC reports if there is a solution
}
```

(5 points)

(b) Consider the following Kripke structure M:



For each of the following formulae φ,

i. check the respective box if 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 $M, s_i \models \varphi$?

φ	CTL	LTL	CTL*	States s_i
$\mathbf{AG}(b)$	☐	☐	☐	
$\mathbf{EG}(b)$	☐	☐	☐	
$\mathbf{EF}(a)$	☐	☐	☐	
$\mathbf{G}(b)$	☐	☐	☐	
$\mathbf{E}[(a \wedge b)\ \mathbf{U}\ (c)]$	☐	☐	☐	

(5 points)

(c) Let ϕ_1, ϕ_2, ϕ_3 be the following CTL* formulae:

$$\begin{aligned} \phi_1 &= \mathbf{EFAG}(p) \\ \phi_2 &= \mathbf{EFG}(p) \\ \phi_3 &= \mathbf{EGF}(p) \end{aligned}$$

For every $i, j \in \{1, 2, 3\}$ such that $i \neq j$, determine whether $\phi_i \models \phi_j$. If so, prove the implication. Otherwise, disprove by giving a Kripke structure M and a state s such that (M, s) satisfies ϕ_i but not ϕ_j.

(5 points)