# 6.0/4.0 VU Formale Methoden der Informatik (185.291)

**December 09, 2011**

| Kennzahl (study id) | Matrikelnummer (student id) | Familienname (family name) | Vorname (first name) | Gruppe (version) |
|---|---|---|---|---|
| | | | | **B** |

**1.)** Consider the following problem:

> **COMPARISON**
>
> INSTANCE: A pair $(\Pi_1, \Pi_2)$ of programs such that:
>
> - $\Pi_1$ takes an integer as input and outputs a string, and
> - $\Pi_2$ takes an integer as input and outputs a string.
>
> It is guaranteed that $\Pi_1$ and $\Pi_2$ terminate on any input integer.
>
> QUESTION: Does there exists an integer $n$ such that $|\Pi_1(n)| > |\Pi_2(n)|$? Here $|J|$ denotes the length of a string $J$, and $\Pi_1(n)$ and $\Pi_2(n)$ are the strings returned by $\Pi_1$ and $\Pi_2$ on input integer $n$, respectively.

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

**(15 points)**

**2.)** (a) Let $\varphi^{uf}$ 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: $\varphi^{uf}$ 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: $FC^E(\varphi^{uf}) = FC^E(\neg\varphi^{uf})$.

**(7 points)**

(b) Answer the following questions and *explain in detail*.

i. Does $\left(FC^E(\varphi^{uf}) \wedge flat^E(\varphi^{uf})\right) \equiv \varphi^{uf}$ hold in general?

ii. Let $\Psi_A(\varphi^{uf})$ be the result of Ackermann's translation applied to $\varphi^{uf}$ and let $\Psi_B(\varphi^{uf})$ be the result of Bryant's translation applied to $\varphi^{uf}$. Does the following hold?

$$\Psi_A(\varphi^{uf}) \text{ is valid} \quad \text{iff} \quad \Psi_B(\varphi^{uf}) \text{ is valid}$$

iii. With the same notation as in ii., does the following hold?

$$\neg\Psi_A(\varphi^{uf}) \text{ is satisfiable} \quad \text{iff} \quad \neg\Psi_B(\varphi^{uf}) \text{ is satisfiable}$$

iv. Consider the sparse method and the procedure which makes a graph chordal. Suppose this procedure introduces $k$ new edges. Is $k$ exponential in the number of vertices of the input graph?

**(8 points)**

**3.)** Compute the weakest precondition of the following program for the postcondition $x = y$.

```
x ← x + y;
if x < 0 then
   abort
else
   while x ≠ y do
      x ← x + 1;
      y ← y + 2
   od
fi
```

**(15 points)**

**4.)** Linear Temporal Logic

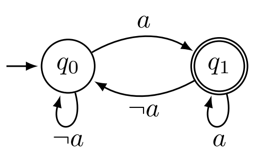
(a) Give an Büchi automaton for the LTL formula $\mathbf{XX}(a \vee \mathbf{FG}b)$.
Since there were two slightly different definitions of Büchi automata used in the lecture slides and in the exercises please provide which definition you are using for your solution:

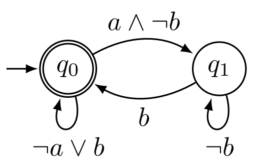
- ☐ [Exercises] A Büchi automaton $\mathcal{A} = (\Sigma, Q, \Delta, I, F)$ is a finite automaton where
  - $\Sigma$ is the finite alphabet,
  - $Q$ is the finite set of states,
  - $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation,
  - $I \subseteq Q$ is the set of initial states, and
  - $F \subseteq Q$ is the set of accepting states.
- ☐ [Lecture Slides] A Büchi automaton $\mathcal{A} = (Q, I, \delta, F, \lambda)$ is a finite automaton where
  - $Q$ is the finite set of states,
  - $I \subseteq Q$ is the set of initial states,
  - $\delta : Q \to 2^Q$ is a transition relation,
  - $F \subseteq Q$ is the set of accepting states, and
  - $\lambda : Q \to 2^P$ is a labeling function where $P$ is the set of propositions.

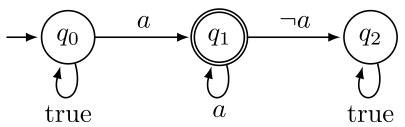
**(4 points)**

(b) For each of the given Büchi automata give a corresponding LTL formula:

**(6 points)**




(c) For this subexercise we define a Büchi-automaton as a 5-tuple $\mathcal{A} = \langle Q, \Sigma, \delta, I, F \rangle$, where

- $Q$ is some finite set of *states*,
- $\Sigma$ is a finite *alphabet*,

- $\Delta \subseteq Q \times \Sigma \times Q$ is the *transition relation*,
- $I \subseteq Q$ is the set of *initial states*, and
- $F \subseteq Q$ is the set of *final states*.

Note, this corresponds to the notion of Büchi-automata as used in the exercises (first option above).

A *word* is an infinite sequence $s_1 s_2 \cdots$ with $s_i \in \Sigma$. A *run* of $\mathcal{A} = \langle Q, \Sigma, \delta, I, F \rangle$ on a word $s_1 s_2 \cdots$ is an infinite sequence $q_0 q_1 \cdots$ of states such that $q_0 \in I$ and $(q_{i-1}, s_i, q_i) \in \Delta$ for all $i \geq 1$. The run $q_0 q_1 \cdots$ is *accepting*, if $q_i \in F$ for infinitely many $i$. An automaton $\mathcal{A}$ *accepts a word*, if it has an accepting run on it.

Assume some fixed Büchi-automaton $\langle Q, \Sigma, \delta, I, F \rangle$ and an infinite word $uv^\omega$ ($uv^\omega = uvvv\cdots$), where $u = s_1 s_2 \cdots s_n$ and $v = s_{n+1} s_{n+2} \cdots s_{2n}$ with $s_i \in \Sigma$ (i.e., the length of $u$ is equal to the length of $v$).

Augment the below C program such that CBMC determines, whether $uv^\omega$ is accepted by a lasso, i.e., if there is a sequence of states $q_0, q_1, \ldots, q_{2n}$ with $q_0 \in I$, $(q_{i-1}, s_i, q_i) \in \Delta$ for all $1 \leq i \leq n$, $q_n = q_{2n}$ and there is some $q_i \in F$ with $n \leq i \leq 2n$. Furthermore, ensure that CBMC reports a lasso in case there exists one. Assume that the states and alphabet symbols are given by natural numbers, i.e., $Q = 1, \ldots, m$ and $\Sigma = 1, \ldots, l$ for some $m, l \in \mathbb{N}$.

```
#define TRUE 1
#define FALSE 0

#define N n  //length of half the input word = length of u = length of v
#define M m  //number of automaton states
#define L l  //number of alphabet symbols

bool delta[M][L][M]; //delta[i][a][j] = TRUE <=> (i,a,j) is in transition relation
bool initial[M];     //initial[i] = TRUE <=> i is initial state
bool final[M];       //final[i] = TRUE <=> i is final state

int word[2N]; //the input word uv

int lasso[2N+1]; // sequence of automaton states

int nondet_int();
```

**(5 points)**