Technische Universität München **Winter term 2010/11**
I7
Prof. J. Esparza / J. Křetínský **10.11.2010**

Automata and Formal Languages – Homework 4

Due 18.11.2010.

Exercise 4.1

Consider the following statement:

> For every regular language L, the minimal DFA recognizing L and the minimal DFA recognizing its complement have the same number of states.

Decide whether this statement holds and give arguments (a counterexample or a proof).

Exercise 4.2

We define the following languages over the alphabet $\Sigma = \{a, b\}$:

- L_1 is the set of all words where between any two occurrences of b's there is at least one a.
- L_2 is the set of all words where every maximal sequence of consecutive a's has odd length.
- L_3 is the set of all words where a occurs only at even positions.
- L_4 is the set of all words where a occurs only at odd positions.
- L_5 is the set of all words of odd length.
- L_6 is the set of all words with an even number of a's.

Remark: For this exercise we assume that the first letter of a nonempty word is at position 1, e.g., $a \in L_4$, $a \notin L_3$.

Your task is to construct an FA, i.e., DFA or NFA or NFA-ε, for

$$L := (L_1 \setminus L_2) \cup \overline{(L_3 \triangle L_4) \cap L_5 \cap L_6} \text{ where } \triangle \text{ denotes the symmetric difference.}$$

while sticking to the following rules:

- You have to start from FAs for $L_1, \ldots, L_6$.
- Any further automaton has to be constructed from already existing automata via an algorithm introduced in the lecture, e.g. Comp, BinOp, UnionNFA, NFAtoDFA, etc.

Try to find an order on the construction steps which yields an FA for L with as few states as possible.

Exercise 4.3

Find a family $\{\mathcal{A}_n\}_{n=1}^{\infty}$ of NFAs with $O(n)$ states such that *every* NFA recognizing the complement of $\mathcal{L}(\mathcal{A}_n)$ has at least 2^n states.

Hint: It is also possible to adapt the standard example $L = \{ww \mid w \in \{a, b\}^*\}$. (Here the complement of L is recognizable by push-down automata, whereas L is not.)

Exercise 4.4

For L_1, L_2 regular languages over an alphabet Σ, the *left quotient* $L_2 \diagdown L_1$ of L_1 by L_2 (note that this is different from the set difference $L_2 \setminus L_1$) is defined by

$$L_2 \diagdown L_1 := \{v \in \Sigma^* \mid \exists u \in L_2 : \ uv \in L_1\}$$

(a) Given finite automata $\mathcal{A}_1, \mathcal{A}_2$, construct an automaton $\mathcal{A}$ such that

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_2) \diagdown \mathcal{L}(\mathcal{A}_1)$$

(b) Is there any difference when taking the *right quotient* $L_1 \diagup L_2 := \{u \in \Sigma^* \mid \exists v \in L_2 : \ uv \in L_1\}$?

(c) Determine the inclusion relation between the following languages:

- L_1
- $(L_1 \diagup L_2).L_2$
- $(L_1.L_2) \diagup L_2$

Exercise 4.5

Check whether the NFA depicted below recognizes Σ^* by means of the algorithm "UnivNFA" presented in the lecture.