Software-Modellierung und Verifikation
Informatik 2
Prof. J.-P. Katoen
RWTH Aachen

Priv.-Doz. T. Noll noll@cs.rwth-aachen.de
Ch. Jansen christina.jansen@cs.rwth-aachen.de

2. Exercise sheet *Compiler Construction 2010*

Due Wed., 03 November 2010, *before* the exercise course begins.

Exercise 2.1: **(2 points)**

Give a NFA $\mathfrak{A}$ (without sink state) with at least two states such that its determinised version $\mathfrak{A}'$ consists of $2^{|\mathfrak{A}|} - 1$ states. Is it possible to give an input word which causes the NFA method seen in lecture 2 to traverse through the whole state space of $\mathfrak{A}'$? If not, modify your automata accordingly.

Exercise 2.2: **(4 points)**

a) For extended matching two principles have been introduced to resolve nondeterminism during analysis, the *longest match* principle and the *first match* principle. Argue why these principles are reasonable to use. Instead, we could have insisted on an unambiguous definition of the symbol classes, i.e. for regular expressions $\alpha_1, \ldots, \alpha_n$ it should hold $\bigcap_{i=1}^{n} [\![\alpha_i]\!] = \emptyset$. Why is this not a good idea from a practical point of view? Give examples to support your explanations.

b) Let $\alpha_1, \ldots, \alpha_n$ be regular expressions over Σ and $w \in \Sigma^*$. In the lecture it was assumed that $\varepsilon \notin [\![\alpha_i]\!] \neq \emptyset$ for all $i \in \{1, \ldots, n\}$. Show that these are reasonable assumptions by proving the following proposition:

- If $[\![\alpha_i]\!] = \emptyset$ for some $i \in \{1, \ldots, n\}$ there exists no *flm*-analysis of w w.r.t. $\alpha_1, \ldots, \alpha_n$ that is not a *flm*-analysis of w w.r.t. $\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_n$ as well.
- If $\varepsilon \in [\![\alpha_i]\!]$ for some $i \in \{1, \ldots, n\}$ then the *flm*-analysis of w w.r.t. $\alpha_1, \ldots, \alpha_n$ is not unique (if it exists).

Exercise 2.3: **(6 points)**

Consider again the propositional logic specified in Ex. 1 and the regular expressions
$\alpha_1 = (sing)^+$
$\alpha_2 = (s + t)(a + \ldots + z)^* g$
$\alpha_3 = tt + ff$
$\alpha_4 = \rightarrow + \vee + \neg + tt + ff$

a) Construct DFAs $\mathfrak{A}_i$ for α_i such that $\mathcal{L}(\mathfrak{A}_i) = [\![\alpha_i]\!]$.

b) Construct a DFA $\mathfrak{A}$ such that $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{A}_1) \cup \mathcal{L}(\mathfrak{A}_2) \cup \mathcal{L}(\mathfrak{A}_3) \cup \mathcal{L}(\mathfrak{A}_4)$.

c) Determine the *first match* partitioning of the set of final states in $\mathfrak{A}$.
(The regular expressions are ordered in an increasing manner.)

d) Compute the (accepting) run of the corresponding backtracking DFA for input $w = singing \vee tt \rightarrow \neg talking$.

e) Give a logic formula for which a decomposition but no longest match decomposition exists. You may extend the regular expressions in order to do so.