# Trial Exam for 02141

May 7 — Spring 2013

## Regular Languages and Finite Automata

### Exercise 1

In this exercise we are going to study a number of languages, $L_1, \cdots, L_7$ over the alphabet $\Sigma = \{a, b\}$. Each one will be desribed as a regular expression, as a DFA, as an NFA, as an $\epsilon$-NFA, as a context free grammar, or as a set describing the set of strings in the language.

For each pairs of languages $(L_i, L_j)$ we are going to determine the relationship between the langauges:

$\subset$ We shall write $L_i \subset L_j$ whenever (1) every string in $L_i$ also is in $L_j$, and (2) there is at least one string in $L_j$ that is not in $L_i$.

$\supset$ We shall write $L_i \supset L_j$ whenever (1) every string in $L_j$ also is in $L_i$, and (2) there is at least one string in $L_i$ that is not in $L_j$.

$=$ We shall write $L_i = L_j$ whenever (1) every string in $L_i$ also is in $L_j$, and (2) every string in $L_j$ also is in $L_i$.

$\Delta$ We shall write $L_i \Delta L_j$ whenever (1) there is at least one string in $L_i$ that is not in $L_j$, and (2) there is at least one string in $L_j$ that is not in $L_i$.

We state without proof, that for each pair of languages $(L_i, L_j)$, exactly one of the above possibilities apply.

Next consider the following seven languages:

$L_1$ The language $L_1$ is described by the context free grammar with productions $S \to aS \mid bS \mid \epsilon$.

$L_2$ The language $L_2$ is described by $\{a^n b^m \mid n \geq 0, m \geq 0\}$.

$L_3$ The language $L_3$ is described by the transition diagram

| | $a$ | $b$ |
|---|---|---|
| $\to \star 1$ | 2 | 3 |
| 2 | 3 | 1 |
| 3 | 3 | 3 |

$L_4$ The language $L_4$ is described by $\{a^n b^n \mid n \geq 0\}$.

$L_5$ The language $L_5$ is described by the regular expression $b{*}a{*}$.

$L_6$ The language $L_6$ is described by the regular expression $(ba){*}$.

$L_7$ The language $L_7$ is described by $\{b^n a^n \mid n \geq 0\}$.

You are required to provide your answer in the form of a table, where rows correspond to the index $i$ and columns correspond to the index $j$, and entries give the classification of the pair $(L_i, L_j)$:

| | $L_1$ | $L_2$ | $L_3$ | $L_4$ | $L_5$ | $L_6$ | $L_7$ |
|---|---|---|---|---|---|---|---|
| $L_1$ | $=$ | | | | | | $\supset$ |
| $L_2$ | | $=$ | | | | | |
| $L_3$ | | | $=$ | | | | |
| $L_4$ | | | | $=$ | | | |
| $L_5$ | | | | | $=$ | | |
| $L_6$ | | | | | | $=$ | |
| $L_7$ | | | | | | | $=$ |

As an example, $L_1 \supset L_7$, and hence there is a $\supset$ in the row for $L_1$ and the column for $L_7$.

## Exercise 2

Consider the following $\epsilon$-NFA over the alphabet $\Sigma = \{a, b, c\}$:

| | $\epsilon$ | $a$ | $b$ | $c$ |
|---|---|---|---|---|
| $\rightarrow \star 1$ | $\{2\}$ | $\{1\}$ | $\emptyset$ | $\emptyset$ |
| $\star 2$ | $\{3\}$ | $\emptyset$ | $\{2\}$ | $\emptyset$ |
| $\star 3$ | $\emptyset$ | $\emptyset$ | $\emptyset$ | $\{3\}$ |

Convert it to a DFA using the "lazy subset construction" (see [HMU06] Section 2.5) and answer the following questions:

**a** How many transitions are there in the resulting DFA?

**b** How many accepting states are there in the resulting DFA?

### Exercise 3

Consider the following DFA over the alphabet $\Sigma = \{a, b\}$:

| | $a$ | $b$ |
|---|---|---|
| $\rightarrow 1$ | 2 | 5 |
| $\star 2$ | 3 | 4 |
| 3 | 2 | 2 |
| 4 | 2 | 2 |
| $\star 5$ | 6 | 7 |
| 6 | 5 | 5 |
| 7 | 5 | 5 |

Construct the minimized DFA using and display it in the same form used above.

## Context-free Languages

### Exercise 4

Consider the Context-Free Grammar $G = (V, T, P, S)$, where

$$\begin{array}{rcl} V & = & \{\mathsf{S}, \mathsf{N}, \mathsf{V}, \mathsf{T}, \mathsf{R}, \mathsf{C}\} \\ T & = & \{\ \texttt{Mama}\ ,\ \texttt{Papa}\ ,\ \texttt{I}\ ,\ \texttt{was}\ ,\ \texttt{am}\ , \\ & & \quad \texttt{the}\ ,\ \texttt{King}\ ,\ \texttt{Queen}\ ,\ \texttt{of}\ , \\ & & \quad \texttt{Mambo}\ ,\ \texttt{Congo}\ ,\ \texttt{Bongo}\ \} \\ S & = & \mathsf{S} \end{array}$$

and $P$ contains the following productions:

$$\begin{array}{rcl} \mathsf{S} & \rightarrow & \mathsf{N}\ \mathsf{V}\ \mathsf{T} \\ \mathsf{N} & \rightarrow & \texttt{Mama} \mid \texttt{Papa} \mid \texttt{I} \\ \mathsf{V} & \rightarrow & \texttt{was} \mid \texttt{am} \\ \mathsf{T} & \rightarrow & \texttt{the}\ \mathsf{R}\ \texttt{of}\ \mathsf{C} \\ \mathsf{R} & \rightarrow & \texttt{King} \mid \texttt{Queen} \\ \mathsf{C} & \rightarrow & \texttt{Mambo} \mid \texttt{Congo} \mid \texttt{Bongo} \end{array}$$

This grammar defines a language, $L(G)$, of (not necessarily correct) statements about yourself and your immediate ancestors.

**a)** Write a leftmost derivation to show that

$$\texttt{Mama was the Queen of Mambo} \in L(G)$$

**b)** Perform a recursive inference (See e.g. [HMU06, Example 5.4]) to show that

$$\texttt{Papa was the King of Congo} \in L(G)$$

**c)** Consider the grammar that extends $G$ into $G' = (V \cup \{\mathsf{Song}\}, T \cup \{\ \texttt{and}\ ,\ \texttt{but}\ \}, P', \mathsf{Song})$, where $P'$ is as $P$ with the addition of

$$\mathsf{Song} \to \mathsf{S} \ \mid\ \mathsf{Song}\ \texttt{and}\ \mathsf{Song} \ \mid\ \mathsf{Song}\ \texttt{but}\ \mathsf{Song}$$

Why is this grammar ambiguous?

### Exercise 5

Now consider the Context-Free Grammar, $G''$, that emerges if you restrict $G$ to consider only the set

$$T' = \{\ \texttt{Mama}\ ,\ \texttt{Papa}\ ,\ \texttt{was}\ ,\ \texttt{the}\ ,\ \texttt{King}\ ,\ \texttt{Queen}\ ,\ \texttt{of}\ ,\ \texttt{Congo}\ \}$$

of terminals, i.e. discard every production where the right hand side contains a symbol not in this set.

**a)** Use the approach described by [HMU06, page 244] to construct a Pushdown Automaton,

$$P_N = (Q, \Sigma, \Gamma, \delta, q_0, Z_0),$$

that accepts by empty stack, i.e. such that $N(P_N) = L(G'')$. How many states and transitions does the resulting PDA have?

**b)** Draw $P_N$ and extend it into a Pushdown Automaton,

$$P_F = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F),$$

that accepts by final state, i.e. such that $L(P_F) = N(P_N)$ (see [HMU06, Section 6.2.3]).

## Semantics

### Exercise 6

Consider the following statement written in the syntax of the **While** language:

$$\texttt{z} := \texttt{1};\ \texttt{while}\ \texttt{y} > \texttt{0}\ \texttt{do}\ (\texttt{y} := \texttt{y} - \texttt{1};\ \texttt{z} := \texttt{z} * \texttt{x})$$

Using the natural semantics for **While**, construct a derivation tree for this statement when executed in a state where `x` has the value **3**, `y` has the value **1**, and `z` has the value **0**. Indicate which rules you have applied by mentioning the appropriate rule name from Table 2.1 of [NN07] for each inference step.

You can use the following abbreviations in your derivations

$$\begin{aligned} \underline{\mathrm{w}} &= \texttt{while}\ \texttt{y} > \texttt{0}\ \texttt{do}\ (\texttt{y} := \texttt{y} - \texttt{1};\ \texttt{z} := \texttt{z} * \texttt{x}) \\ S_0 &= \texttt{y} := \texttt{y} - \texttt{1};\ \texttt{z} := \texttt{z} * \texttt{x} \end{aligned}$$

and you can write $s_{ijk}$ for the state $[\texttt{x} \mapsto i, \texttt{y} \mapsto j, \texttt{z} \mapsto k]$.

## Exercise 7

Let $s$ and $s'$ be two states satisfying that $s\ x = s'\ x$ for all $x \in FV(a)$. Prove that $\mathcal{A}[\![a]\!]s = \mathcal{A}[\![a]\!]s'$.

## Exercise 8

Consider the following statement $S$ written in the syntax of the **While** language extended with the parallel construct `par`:

$$(\mathtt{x} := \mathtt{x} + \mathtt{1};\ \mathtt{x} := \mathtt{x} + \mathtt{1})\ \mathtt{par}\ (\mathtt{while}(\mathtt{x} < 4)\ \mathtt{do}\ (\mathtt{x} := \mathtt{x} * 2))$$

We assume the statement is executed in a state where $x$ has the value 1.

1. Construct a derivation sequence in structural operational semantics showing that the execution of the statement can terminate in the same state where $x$ has the value 6.
2. Construct a derivation tree in natural semantics showing that the execution of the statement can terminate in the same state where $x$ has the value 5.
3. State all other possible results of the execution of the statement (you don't have to provide derivation sequences).
4. Is this statement deterministic? Why?

*Hint.* Always indicate which rules you have applied by mentioning the appropriate rule name(s) for each derivation step.

## Exercise 9

Consider the following statement, written in the syntax of the language **Proc** that extends **While** with blocks and procedure declarations (see [NN07] Section 3.2).

```
     begin
a      var x := 0;
b      proc p is x := x + 1;
c      proc q is (call p; y := x + 2);
d      proc r is (begin (proc p is x := x * 2); call q end)
       begin
e        var x := 3;
         call r;
         call p;
f        z := x + y
       end
     end
```

Each assignment in the program corresponds to a letter denoting the line on which it occurs. For example, `x := 0` corresponds to line $a$, `x := x + 1` corresponds to line $b$, and so on.

We assume that the statement is executed in a state where the variables `x`, `y`, and `z` have the value **0**. We are interested in the sequence of assignments and the value of the different variables when the statement is executed using

**a** dynamic scope rules for variables as well as procedures

**b** dynamic scope rules for variables and static scope rules for procedures

**c** static scope rules for variables as well as procedures

For questions **a** and **b** you are required to provide your answer in the form of a table where the first row corresponds to the sequence of assignments taken, and the remaining three rows correspond to the values of the different variables *after* the respective assignment has taken place. For example, the table for question **a** starts out as follows:

| | $a$ | $e$ | ... | | | | | | ... |
|---|---|---|---|---|---|---|---|---|---|
| x | 0 | 3 | ... | | | | | | |
| y | 0 | 0 | ... | | | | | | |
| z | 0 | 0 | ... | | | | | | ... |

For question **c** it suffices to provide the value of the variable `z` after termination of the program.

## References


[HMU06] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley, 2006.

[NN07] H. Riis Nielson and F. Nielson. *Semantics with Applications: An Appetizer*. Undergraduate Topics in Computer Science. Springer, 2007.