Fundamentals of theory of computation I

Test 2

Send your solutions as a single NEPTUNCODE.zip file to ktichler@inf.elte.hu or tichlerk@gmail.com

---

Exercise 1: Make a reduced grammar, according to the algorithm we learnt, generating the same language as the following type 2 grammar G. $G = \langle \{S, A, B, C, D, E\}, \{a, b\}, S, P \rangle$, where the set of production rules P is a follows.

$$\begin{aligned}
&S \to AaA \,|\, DaD\\
&A \to SbDD \,|\, a\\
&B \to aAE \,|\, BC\\
&C \to \lambda \,|\, SA\\
&D \to SbSA \,|\, B\\
&E \to aBSCC
\end{aligned}$$

Exercise 2: Make a grammar in Chomsky normal form, according to the algorithm we learnt, generating the same language as the following type 2 grammar G. $G = \langle \{S, A, B\}, \{a, b\}, S, P \rangle$, where the set of production rules P is a follows.

$$\begin{aligned}
&S \to AA \,|\, AaBb\\
&A \to aB \,|\, \lambda\\
&B \to BaS \,|\, b
\end{aligned}$$

Exercise 3: Decide by *CYK algorithm* whether the word *abcab* can be generated in the grammar $G = \langle \{S, A, B, C, K\}, \{a, b, c\}, S, P \rangle$, where the set of production rules P is a follows.

$$\begin{aligned}
&S \to SS \,|\, BC \,|\, SB\\
&A \to KK \,|\, a\\
&B \to AB \,|\, b\\
&C \to SB \,|\, c\\
&K \to KB \,|\, c
\end{aligned}$$

Exercise 4: Let $|u|_t$ denote the number of t's in the word u. Make a pushdown automaton accepting the language $L = \{\, a^n b^k \,|\, n \geq k \geq 1 \,\}$.

You are allowed to accept L either by accepting states or by empty store. The default is accepting by accepting states. State it clearly, if your solution accepts L by empty store.

Choose EXACTLY ONE from the following two exercises!

Exercise 5A: Let $|u|_t$ denote the number of t's in the word u. Prove by Myhill-Nerode Theorem, that $\Big\{u \in \{a, b\}^* \,\Big|\, |u|_a = |u|_b + 2\Big\} \notin \mathcal{L}_3$.

OR

Exercise 5B: Prove by Bar-Hillel Lemma, that $\{(b^n a^n)^n \,|\, n \in \mathbb{N}\} \notin \mathcal{L}_2$.

Exercise 1: Make a reduced grammar, according to the algorithm we learnt, generating the same language as the following type 2 grammar G. $G = \langle \{S, A, B, C, D, E\}, \{a, b\}, S, P \rangle$, where the set of production rules P is a follows.

$$
\begin{aligned}
&S \to AaA \mid DaD \\
&A \to SbDD \mid a \\
&B \to aAE \mid BC \\
&C \to \lambda \mid SA \\
&D \to SbSA \mid B \\
&E \to aBSCC
\end{aligned}
$$

Step 1.

$A_1 = \{A, C\}$

$A_2 = \{A, C, S\}$

$A_3 = \{A, C, S, D\} = A_4$ = set of active nonterminals

Leave all

$$
\begin{aligned}
&S \to AaA \mid DaD \\
&A \to SbDD \mid a \\
&C \to \lambda \mid SA \\
&D \to SbSA
\end{aligned}
$$

$R_1 = \{S\}$

$R_2 = \{S, A, D\}$ = set of reachable nonterminals

$$
\begin{aligned}
&S \to AaA \mid DaD \\
&A \to SbDD \mid a \\
&D \to SbSA
\end{aligned}
$$

Exercise 2: Make a grammar in Chomsky normal form, according to the algorithm we learnt, generating the same language as the following type 2 grammar G. $G = \langle \{S, A, B\}, \{a, b\}, S, P \rangle$, where the set of production rules P is a follows.

$$
\begin{aligned}
&S \to AA \mid AaBb \\
&A \to aB \mid \lambda \\
&B \to BaS \mid b
\end{aligned}
$$

Step 1: introducing fake terminals

$$
\begin{aligned}
&S \to AA \mid AXBY \\
&A \to XB \mid \lambda \\
&B \to BXS \mid b \\
&X \to a \\
&Y \to b
\end{aligned}
$$

Step 2: eliminating λ rules

$U_1 = \{A, B\}$

$U_2 = \{A, B, S\} = U_3 = U$

$S \rightarrow AA \mid A \mid AXBY \mid XBY \mid AXY \mid XY$

$A \rightarrow XB \mid X$

$B \rightarrow BXS \mid XS \mid BX \mid X \mid b$

$X \rightarrow a$

$Y \rightarrow b$

Step 3: Decreasing the length of the rules

$S \rightarrow AA \mid A \mid AZ_1 \mid XZ_2 \mid AZ_3 \mid XY$

$A \rightarrow XB \mid X$

$B \rightarrow BZ_4 \mid XS \mid BX \mid X \mid b$

$X \rightarrow a$

$Y \rightarrow b$

$Z_1 \rightarrow XZ_2$

$Z_2 \rightarrow BY$

$Z_3 \rightarrow XY$

$Z_4 \rightarrow XS$

Step 4: Eliminating chain rules

$U_1(A) = \{A\}, \quad U_2(A) = \{S, A\} = U(A)$

$S \rightarrow AA \mid XB \mid X \mid AZ_1 \mid XZ_2 \mid AZ_3 \mid XY$

$A \rightarrow XB \mid X$

$B \rightarrow BZ_4 \mid XS \mid BX \mid X \mid b$

$X \rightarrow a$

$Y \rightarrow b$

$Z_1 \rightarrow XZ_2$

$Z_2 \rightarrow BY$

$Z_3 \rightarrow XY$

$Z_4 \rightarrow XS$

Exercise 3: Decide by *CYK algorithm* whether the word *abcab* can be generated in the grammar $G = \langle \{S, A, B, C, K\}, \{a, b, c\}, S, P \rangle$, where the set of production rules P is a follows.

$$
\begin{aligned}
&S \to SS \,|\, BC \,|\, SB \\
&A \to KK \,|\, a \\
&B \to AB \,|\, b \\
&C \to SB \,|\, c \\
&K \to KB \,|\, c
\end{aligned}
$$

$\{S, C\}$				
$\{\}$	$\{S, C\}$			
$\{S\}$	$\{\}$	$\{K\}$		
$\{B\}$	$\{S\}$	$\{\}$	$\{B\}$	
$\{A\}$	$\{B\}$	$\{C, K\}$	$\{A\}$	$\{B\}$
a	b	c	a	b

Since $H_{1,5}$ contains S, abcab can be derived in the grammar.

Exercise 4: Let $|u|_t$ denote the number of t's in the word u. Make a pushdown automaton accepting the language $L = \{\, a^n b^k \,|\, n \geq k \geq 1 \,\}$.

You are allowed to accept L either by accepting states or by empty store. The default is accepting by accepting states. State it clearly, if your solution accepts L by empty store.

$\langle \{\#, a, b\}, \{q_0, q_1, q_2\}, \{a, b\}, M, q_0, \#, \{q_2\})$

$\# q_0 a \to \# a q_0$

$a q_0 a \to a a q_0$

$a q_0 b \to q_1$

$a q_1 \to q_1$

$a q_1 b \to q_1$

$\# q_1 \to q_2$

e.g. aaabb

$\# q_0 aaabb \Rightarrow \# a q_0 aabb \Rightarrow \# aa q_0 abb \Rightarrow \# aaa q_0 bb \Rightarrow \# aa q_1 b$

$\Rightarrow \# a q_1 \Rightarrow \# q_1 \Rightarrow q_2$

Exercise 5A: Let $|u|_t$ denote the number of t's in the word u. Prove by Myhill-Nerode Theorem, that $\left\{u \in \{a,b\}^* \,\middle|\, |u|_a = |u|_b + 2\right\} \notin \mathcal{L}_3$.

if $b^k \in L_{a^n}$, so $k < n$

L_{a^n} and L_{a^m} $(n > m)$ are different

Because $b^{n-1} \in L_{a^n}$, we ~~know~~ $a^n b^{n-2}$ has less b than a,

$b^{n-2} \notin L_{a^m}$, because $n-2 \geq m$, $a^m b^{n-2}$ doesn't have less b than a.

All, by Myhill-Nerode Theorem, $\{u \in \{a,b\}^* \mid |u|_a = |u|_b + 2\} \notin \mathcal{L}_3$.