5 Context-Free Grammars

For the derivability relation we use notation $\underset{G}{\Rightarrow}$, $\overset{*}{\underset{G}{\Rightarrow}}$ instead of $\underset{G}{\rightarrow}$, $\overset{*}{\underset{G}{\rightarrow}}$ as used in [Kozen]. The former is more popular.

5.1 Consider the CFG $G = (\{E, T, F\}, \{a, b, c, +, -, \cdot, /, (,)\}, P, E)$, where P comprises the productions

$$\begin{aligned}
&E \rightarrow T \mid E + T \mid E - T\\
&T \rightarrow F \mid T \cdot F \mid T/F\\
&F \rightarrow a \mid b \mid c \mid (E)
\end{aligned}$$

Find the derivation trees for the following strings.

a) $a \cdot b + c$

b) $a + a - b \cdot (a/b + b/c)$

5.2 Find CFGs which generate the following languages.

a) All strings in $\{0, 1\}^*$ for which every 0 is followed by 1 immediately to the right.

b) All strings in $\{0, 1\}^*$ which are palindromes.

c) $\{0^n 1^n \mid n \geq 0\}$

d) All string in $\{a, b\}^*$ containing at least one a and one b, such that the number of a's preceding the first b is the same as the number of b's following the last a.

5.3 Consider the CFG $G = (\{S, A, B\}, \{a, b\}, P, S)$, where P comprises the productions

$$\begin{aligned}
&S \rightarrow aB \mid bA\\
&A \rightarrow a \mid aS \mid bAA\\
&B \rightarrow b \mid bS \mid aBB
\end{aligned}$$

Show that G is ambiguous.

5.4 For a CFG $G = (N, \Sigma, P, S)$, a symbol X is *useful* if there exists a derivation $S \overset{*}{\underset{G}{\Rightarrow}} \alpha X \beta \overset{*}{\underset{G}{\Rightarrow}} w$ where $w \in \Sigma^*$. Otherwise X is *useless*. So a useless symbol does not occur in any derivation of a terminal string from S.

a) Let G be a CFG consisting of the following productions (S is the start symbol):

$$\begin{aligned}
&S \rightarrow AB \mid CA\\
&A \rightarrow a\\
&B \rightarrow BC \mid AB\\
&C \rightarrow aB \mid b
\end{aligned}$$

Find an equivalent CFG (i.e. a grammar which generates the same language) without useless nonterminal symbols.

This can be done by first finding each nonterminal from which no terminal string can be generated. All the productions containing such nonterminals can be removed. Then one finds those nonterminals that do not occur in any sentential form and removes the productions containing them. For details see [Hopcroft&Ullman].

b) In the algorithm outlined above, the order of the two steps is important. Find a CFG for which reversing this order results in a grammar with some remaining useless symbols.

5.5 Let G be a CFG consisting of the following productions (S is the start symbol):

$$\begin{aligned}&S \to AB\\&A \to SA \mid BB \mid bB\\&B \to b \mid aA \mid \epsilon\end{aligned}$$

Find an equivalent CFG with a single ϵ-production $S \to \epsilon$, and without unit productions.

5.6 Find equivalent Chomsky normal-form CFGs for the two CFGs below (S is the start symbol in both cases).

a) $S \to \neg S \mid (S \supset S) \mid p \mid q$

b)
$$\begin{aligned}&S \to A \mid ABA\\&A \to aA \mid B \mid a\\&B \to bB \mid b\end{aligned}$$

5.7 [This is outside of the present scope of the course]. Find equivalent Greibach normal-form CFGs for the two CFGs below (S is the start symbol in both cases).

a)
$$\begin{aligned}&S \to AA \mid 0\\&A \to SS \mid 1\end{aligned}$$

b)
$$\begin{aligned}&S \to AS \mid AB\\&A \to BS \mid a\\&B \to AA \mid b\end{aligned}$$

5.2 S is the start symbol in all grammars below.

a) $S \to 1S \mid 01S \mid \epsilon$

b) $S \to \epsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1$

c) $S \to 0S1 \mid \epsilon$

d) $S \to aSb \mid ab \mid bAa$
$A \to \epsilon \mid aA \mid bA$

Justification: Any string over $\{a, b\}$ can be generated from A. Productions $S \to ab \mid bAa$ generate the first b and the last a (and the number of a's preceding the first b is the same as the number of b's following the last a, it is 1 for the first production and 0 for the second). Production $S \to aSb$ adds one a preceding the first b and one b following the last a.

5.3 The string $aabbab$ has two distinct left derivations:
$S \Rightarrow aB \Rightarrow aaBB \Rightarrow aabSB \Rightarrow aabbAB \Rightarrow aabbaB \Rightarrow aabbab$
$S \Rightarrow aB \Rightarrow aaBB \Rightarrow aabB \Rightarrow aabbS \Rightarrow aabbaB \Rightarrow aabbab$

5.4 a) No terminal string can be derived from B. Thus all productions involving B either on the left hand side or on the right hand side may be removed. This gives:

$S \to CA$
$A \to a$
$C \to b$

Since both A and C can occur in derivations of terminal strings from the start symbol, there remain no useless symbols in the above grammar.

b) For instance take

$$S \to AB \mid a$$
$$A \to a$$

1. No terminal string can be derived from B, so B is useless. Remove $S \to AB$. 2. Now no string containing A can be derived from S, so A is useless. Remove $A \to a$.

Doing step 2 first does not discover any useless symbol. Performing then step 1 we remove $S \to AB$ only. A is not found useless.

5.5 The proof of Lemma 21.3 in [Kozen] suggests a method of removing ϵ- and unit productions from a CFG $G = (N, \Sigma, P, S)$. First we add productions to P in order to obtain the smallest $P_1 \supseteq P$ such that

(a) if $A \to \alpha B \beta$ and $B \to \epsilon$ are in P_1 then $A \to \alpha\beta$ is in P_1.

Any nonempty terminal string derived from S in G can be derived in (N, Σ, P_1, S) without using any ϵ-production. So we can remove the ϵ-productions from P_1, obtaining P_1'.

Now we add productions to P_1' in order to obtain the smallest $P_2 \supseteq P_1'$ such that

(b) if $A \to B$ and $B \to \gamma$ are in P_2 then $A \to \gamma$ is in P_2.

Any terminal string derived from S in (N, Σ, P_1', S) can be derived in (N, Σ, P_2, S) without using any unit production. Thus we can remove the unit productions from P_2, obtaining P_2'.

$G' = (N, \Sigma, P_2', S)$ is the result, $L(G') = L(G) - \{\epsilon\}$. (Notice that in [Kozen] the rules (a) and (b) are applied together. Doing this separately, as above, is also correct.)

For the given grammar new productions are added as follows. In order to remove ϵ-productions:

production	with	production	gives	production
$B \to \epsilon$		$S \to AB$		$S \to A$
		$A \to BB$		$A \to B$
		$A \to bB$		$A \to b$
		$A \to B$		$A \to \epsilon$
$A \to \epsilon$		$S \to AB$		$S \to B$
		$A \to SA$		$A \to S$
		$B \to aA$		$B \to a$
		$S \to A$		$S \to \epsilon$
$S \to \epsilon$		$A \to SA$		$A \to A$
		$A \to S$		$A \to \epsilon$
$B \to \epsilon$		$S \to B$		$S \to \epsilon$

The obtained set P_1' of productions is:

$$\begin{aligned} &S \to AB \mid A \mid B \\ &A \to BB \mid B \mid bB \mid b \mid SA \mid S \\ &B \to aA \mid a \mid b \end{aligned}$$

To get rid of unit productions:

production	with	production	gives	production
$S \to A$		$A \to BB$		$S \to BB$
		$A \to B$		$S \to B$
		$A \to bB$		$S \to bB$
		$A \to b$		$S \to b$
		$A \to SA$		$S \to SA$
		$A \to S$		$S \to S$
$S \to B$		$B \to aA$		$S \to aA$
		$B \to a$		$S \to a$
		$B \to b$		$S \to b$
$A \to B$		$B \to aA$		$A \to aA$
		$B \to a$		$A \to a$
		$B \to b$		$A \to b$
$A \to S$		$S \to AB$		$A \to AB$
		$S \to A$		$A \to A$
		$S \to B$		$A \to B$

The obtained set P_2' of productions is:

$$
\begin{aligned}
S &\to AB \mid BB \mid bB \mid b \mid SA \mid aA \mid a \\
A &\to AB \mid BB \mid bB \mid b \mid SA \mid aA \mid a \\
B &\to aA \mid a \mid b
\end{aligned}
$$

As we want to obtain a grammar equivalent to the initial one, the removed production $S \to \epsilon$ has to be added.

5.6 a) Introduce productions for each terminal symbol which does not occur on its own on the right hand side of some production, i.e.:

$$
\begin{aligned}
&A \to \neg \\
&B \to (\\
&C \to \supset \\
&D \to)
\end{aligned}
$$

Then replace all such terminal symbols in the original grammar with the corresponding nonterminal from the productions above.

$$
\begin{aligned}
&S \to AS \mid BSCSD \mid p \mid q \\
&A \to \neg \\
&B \to (\\
&C \to \supset \\
&D \to)
\end{aligned}
$$

The only production above which is not in Chomsky normal-form is $S \to BSCSD$. We can systematically rewrite this production into a set of productions in Chomsky normal-form as follows:

$S \to BSCSD$ is replaced by $S \to BE$ and $E \to SCSD$
$E \to SCSD$ is replaced by $E \to SF$ and $F \to CSD$
$F \to CSD$ is replaced by $R \to CG$ and $G \to SD$

Thus an equivalent Chomsky normal-form grammar is obtained:

$$
\begin{aligned}
&S \to AS \mid BE \mid p \mid q \\
&E \to SF \\
&F \to CG \\
&G \to SD \\
&A \to \neg \\
&B \to (\\
&C \to \supset \\
&D \to)
\end{aligned}
$$

b) First eliminate all unit productions. This yields

$$
\begin{aligned}
&S \to ABA \mid aA \mid a \mid bB \mid b \\
&A \to aA \mid a \mid bB \mid b \\
&B \to bB \mid b
\end{aligned}
$$

We then proceed as in the previous exercise: productions for terminal symbols are introduced where necessary and productions with right hand sides comprising three or more nonterminals are systematically rewritten into a set of productions in Chomsky normal-form. This results in the following grammar:

$$
\begin{aligned}
&S \to AE \mid CA \mid DB \mid a \mid b \\
&A \to CA \mid DB \mid a \mid b \\
&B \to DB \mid b \\
&E \to BA \\
&C \to a \\
&D \to b
\end{aligned}
$$

5.7 a) We follow the method shown in [Hopcroft&Ullman]. Since the grammar already is in Chomsky normal-form, some work is saved. First we rename the nonterminals as follows: $S = A_1$ and $B = A_2$. This yields the grammar:

$$
\begin{aligned}
&A_1 \to A_2A_2 \mid 0 \\
&A_2 \to A_1A_1 \mid 1
\end{aligned}
$$

Then we inspect each production for A_1 and A_2, and construct new rules as follows. Suppose $A_i \to A_j\alpha$ is a production (where α is a string of nonterminals). If $i < j$, then that production is left as it is. If $i = j$, then we must do something to eliminate this left recursion. This is described below. If $i > j$, then we replace the production by the set of productions obtained by replacing A_j in the rule $A_i \to A_j\alpha$ by the right hand side of each production for A_j.

$A_1 \to A_2A_2$ is left unchanged.
$A_2 \to A_1A_1$ is replaced by $A_2 \to A_2A_2A_1 \mid 0A_1$

Here, $A_2 \to A_2A_2A_1$ is an example of a left-recursive production. In order to get rid of such productions, we proceed as follows. Suppose

$$A \to A\alpha_1 \mid \cdots \mid A\alpha_n \mid \beta_1 \mid \cdots \mid \beta_m$$

are all the productions for A, where $\alpha_1, \ldots, \alpha_n$ are strings of nonterminals, $\beta_1, \ldots, \beta_m$ are strings of terminals and nonterminals which do not begin by A. The A production are now replaced by

$$A \to \beta_1 \mid \cdots \mid \beta_m \mid \beta_1A' \mid \cdots \beta_mA'$$

where A' is a new nonterminal. The productions for A' are

$$A' \to \alpha_1 \mid \cdots \mid \alpha_n \mid \alpha_1A' \mid \cdots \mid \alpha_nA'$$

Thus $A_2 \to A_2A_2A_1 \mid 0A_1 \mid 1$ should be replaced by

$$
\begin{aligned}
&A_2 \to 0A_1 \mid 1 \mid 0A_1A_3 \mid 1A_3 \\
&A_3 \to A_2A_1 \mid A_2A_1A_3
\end{aligned}
$$

Then we continue with the rules for A_3.

$A_3 \to A_2A_1$ is replaced by
  $A_3 \to 0A_1A_1 \mid 1A_1 \mid 0A_1A_3A_1 \mid 1A_3A_1$
$A_3 \to A_2A_1A_3$ is replaced by
  $A_3 \to 0A_1A_1A_3 \mid 1A_1A_3 \mid 0A_1A_3A_1A_3 \mid 1A_3A_1A_3$

Thus the following grammar has been obtained:

$$
\begin{aligned}
&A_1 \to A_2A_2 \mid 0\\
&A_2 \to 0A_1 \mid 1 \mid 0A_1A_3 \mid 1A_3\\
&A_3 \to 0A_1A_1 \mid 1A_1 \mid 0A_1A_3A_1 \mid 1A_3A_1\\
&\qquad \mid 0A_1A_1A_3 \mid 1A_1A_3 \mid 0A_1A_3A_1A_3 \mid 1A_3A_1A_3
\end{aligned}
$$

The result of all our efforts is that whenever the first symbol of a right hand side of a production is a nonterminal, it will have a higher number than the nonterminal on the left hand side. However, the grammar is still not quite in Greibach normal-form. In order to obtain Greibach normal-form, we have to do one more pass over the productions. Whenever the first symbol on a right hand side is a nonterminal, it should be replaced by the right hand sides of the productions for this nonterminal.

$A_1 \to A_2A_2$ is replaced by
  $A_1 \to 0A_1A_2 \mid 1A_2 \mid 0A_1A_3A_2 \mid 1A_3A_2$

Thus we finally obtain:

$$
\begin{aligned}
&A_1 \to 0A_1A_2 \mid 1A_2 \mid 0A_1A_3A_2 \mid 1A_3A_2 \mid 0\\
&A_2 \to 0A_1 \mid 1 \mid 0A_1A_3 \mid 1A_3\\
&A_3 \to 0A_1A_1 \mid 1A_1 \mid 0A_1A_3A_1 \mid 1A_3A_1\\
&\qquad \mid 0A_1A_1A_3 \mid 1A_1A_3 \mid 0A_1A_3A_1A_3 \mid 1A_3A_1A_3
\end{aligned}
$$