Introduction to Language Theory and Compilation: Exercises

Session 3: Introduction to grammars

Definition

A grammar is a tuple $G = \langle V, T, P, S \rangle$ where

- V is the set of *nonterminals* (or *variables*);
- T is the set of *terminals*;
- P is the set of *production rules*. In the general case:

$$P \subseteq \underbrace{(V \cup T)^* V (V \cup T)^*}_{\text{at least one variable}} \times (V \cup T)^*$$

- $S \in V$ is the *start symbol*.

Example

Let G be a grammar whose production rules in P are:

$$\begin{array}{lcl} S & \rightarrow & aSbS \\ S & \rightarrow & bSaS \\ S & \rightarrow & e \end{array}$$

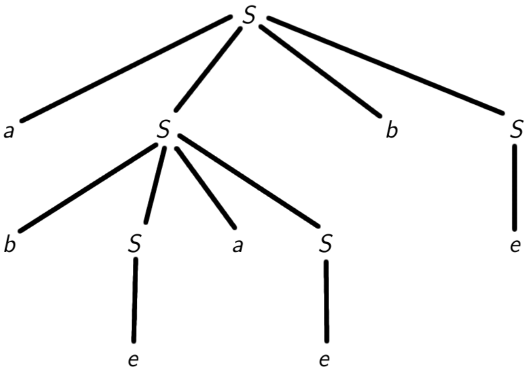
The rules can also be written in a more compact form:

$$S \rightarrow aSbS \mid bSaS \mid e$$

S is the only variable (or nonterminal) and is also the start symbol. We have $T = \{a, b, e\}$.

Parse tree (or derivation tree)

The string `abeaebe` can be parsed using G and is thus part of the language defined by the grammar.



Class 0: Unrestricted grammars
No restrictions on the structure of production rules.

$\Longrightarrow$ A bag of production rules.

Chomsky hierarchy – class 1/3

Class 1: Context-sensitive grammars Every production rule must have the following structure:

$$\alpha \rightarrow \beta$$

with $|\alpha| \leqslant |\beta|$. As an exception, the following rule may be part of the grammar as well:

$$S \rightarrow \varepsilon$$

where S is the start symbol. This rule is only allowed if S never appears on the right hand of any production rule.

$\Longrightarrow$ A starting production rule S and each production rule is composed of a pair of ordered sets (left,right) where $|\ \mathit{left}\ | \leqslant |\ \mathit{right}\ |$.

Class 2: Context-free grammars (CFG) Every rule must obey the following structure:

$$A \rightarrow \alpha$$

$\Longrightarrow$ A starting production rule S and each production rule is composed of a pair of ordered sets (left,right) where $\mid left \mid = 1$

Chomsky hierarchy – class 3/3

Class 3: Regular grammars Two subclasses:

Right linear grammars Rules must obey this structure:

$$A \to wB \quad \text{or} \quad A \to w \qquad (w \in T^*)$$

Left linear grammars Rules must obey this structure:

$$A \to Bw \quad \text{or} \quad A \to w \qquad (w \in T^*)$$

$\Longrightarrow$ A starting production rule S and each production rule is composed of a pair of ordered sets (left,right) where $\mid left \mid = 1$ and *right* may contain **some terminals** but at most **one starting/ending variable**.

Chomsky – classes summary

Class 0: Unrestricted grammars A bag of production rules

Class 1: CS grammars A starting production rule S and each production rule is composed of a pair of ordered sets (left,right) where $\mid left \mid \leqslant \mid right \mid$.

Class 2: CF grammars A starting production rule S and each production rule is composed of a pair of ordered sets (left,right) where $\mid left \mid = 1$

Class 3: RE grammars A starting production rule S and each production rule is composed of a pair of ordered sets (left,right) where $\mid left \mid = 1$ and *right* may contain some terminals but at most one starting/ending variable

Exercise 1

Informally describe the languages generated by the following grammars and also specify what kind of grammars they are:

(a)
$$
\boxed{\begin{array}{rcl}
S & \rightarrow & abcA \\
 & & Aabc \\
A & \rightarrow & \varepsilon \\
Aa & \rightarrow & Sa \\
cA & \rightarrow & cS
\end{array}}
$$

(b)
$$
\boxed{\begin{array}{rcl}
S & \rightarrow & 0 \\
 & & 1 \\
 & & 1S
\end{array}}
$$

(c)
$$
\boxed{\begin{array}{rcl}
S & \rightarrow & a \\
 & & *SS \\
 & & +SS
\end{array}}
$$

Solution for exercise 1

1. Unrestricted grammar giving all strings made of *abc* taken at least once. Not CS because $A \rightarrow \epsilon$ and the only rule which can generate ϵ is S.
2. Right linear grammar giving all strings made of 1s which may be followed by a single 0. 0 itself is also accepted.
3. Context-free grammar giving all arithmetical expressions (in polish notation) using addition and multiplication with the mathematical variable called a. Not class 3 because two variables $(2 \times S)$ in the set *right*.

Exercise 2

Let G be the following grammar:

$$
\boxed{\begin{array}{lcl}
S & \rightarrow & AB \\
A & \rightarrow & Aa \\
A & \rightarrow & bB \\
B & \rightarrow & a \\
B & \rightarrow & Sb
\end{array}}
$$

1. Is G a regular grammar?
2. Give the *parse tree* for
   a) $baabaab$
   b) $bBABb$
   c) $baSb$
3. Give the *leftmost* and *rightmost derivations* for $baabaab$

Solution for exercise 2 – 1/5

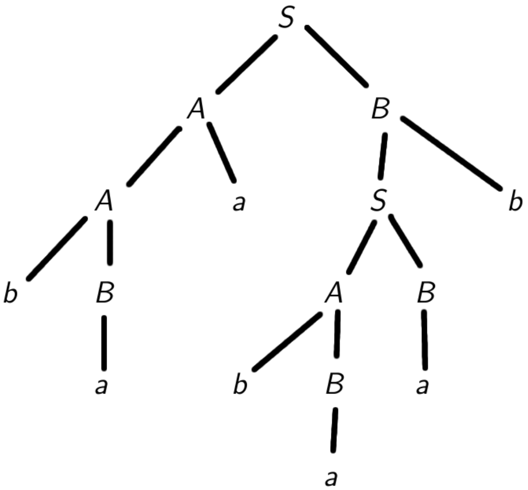
1. Is G regular?

The given grammar is context-free but *not regular* : there cannot be a rule like $A \rightarrow \alpha B$ combined with a rule like $A \rightarrow B\alpha$ in a regular grammar.

Left and **Right** linear grammars are exclusive.

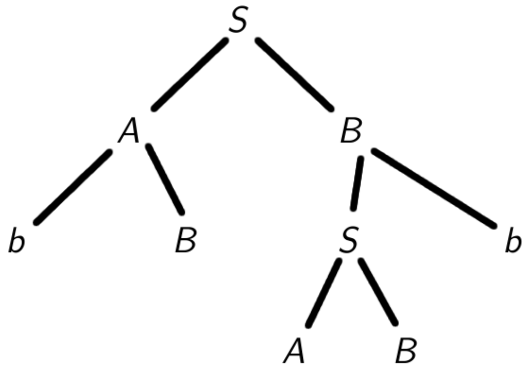
Solution for exercise 2 – 2/5

2. Give the *parse tree* for
   a) *baabaab*



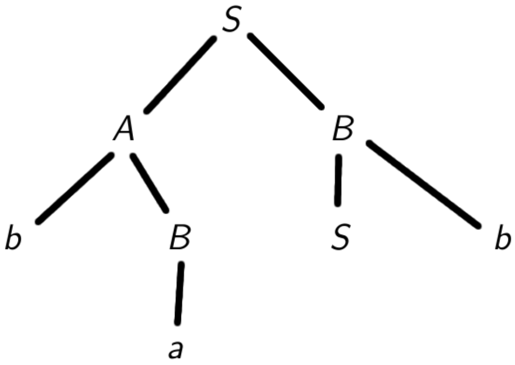
Solution 2 for exercise 2 – 3/5

2. Give the *parse tree* for
   b) $bBABb$



Solution for exercise 2 – 4/5

2. Give the *parse tree* for
   c) *baSb*



3. Give the *leftmost* and *rightmost derivations* for *baabaab*

The *leftmost* derivation of *baabaab* is:
$S \Rightarrow AB \Rightarrow AaB \Rightarrow bBaB \Rightarrow baaB \Rightarrow baaSb \Rightarrow baaABb \Rightarrow baabBBb \Rightarrow baabaBb \Rightarrow baabaab$

The *rightmost* derivation is: $S \Rightarrow AB \Rightarrow ASb \Rightarrow AABb \Rightarrow AAab \Rightarrow AbBab \Rightarrow Abaab \Rightarrow Aabaab \Rightarrow bBabaab \Rightarrow baabaab$

Exercise 3

3. Write a context-free grammar that generates all strings of *a*s and *b*s (in any order) such that there are more *a*s than *b*s. Test your grammar on the input *baaba* by giving a derivation.

Exercise 3

3. Write a context-free grammar that generates all strings of as and bs (in any order) such that there are more as than bs. Test your grammar on the input *baaba* by giving a derivation.

Trick: *when you add a b, you also add a a in order to have at least* $|a| = |b|$

Solution for exercise 3

$$S \rightarrow bSa \mid aSb \mid abS \mid baS \mid Sab \mid Sba \mid Sa \mid aS \mid a$$

We can derive *baaba* from this grammar:

$$S \Rightarrow baS \Rightarrow baabS \Rightarrow baaba$$

Exercise 4

4. Write a context-sensitive grammar that generates all strings of as, bs and cs (in any order) such that there are as many of each. Give a derivation of *cacbab* using your grammar.

Solution for exercise 4

n^o	Rule			idea
(1)	S	$\rightarrow$	$S\prime$	words
(2)	S	$\rightarrow$	ε	nowords
(3)	$S\prime$	$\rightarrow$	ABC	$\mid a \mid = \mid b \mid = \mid c \mid$
(4)	$S\prime$	$\rightarrow$	$ABCS\prime$	*concat words*
(5)	AB	$\rightarrow$	BA	*swap a and b*
(6)	AC	$\rightarrow$	CA	*swap a and a*
(7)	BA	$\rightarrow$	AB	*swap b and a*
(8)	BC	$\rightarrow$	CB	*swap b and c*
(9)	CA	$\rightarrow$	AC	*swap c and a*
(10)	CB	$\rightarrow$	BC	*swap c and b*
(11)	A	$\rightarrow$	a	*variable; to terminal*
(12)	B	$\rightarrow$	b	*variable; to terminal*
(13)	C	$\rightarrow$	c	*variable; to terminal*

cacbab can be derived from this grammar:

$$S \overset{(1)}{\Rightarrow} S' \overset{(4)}{\Rightarrow} ABCS' \overset{(3)}{\Rightarrow} ABCABC \overset{(8)}{\Rightarrow} ACBABC \overset{(6)}{\Rightarrow} CABABC \overset{(8)}{\Rightarrow} CABACB \overset{(6)}{\Rightarrow} CABCAB \overset{(8)}{\Rightarrow} CACBAB \overset{(13)}{\Rightarrow} cACBAB \overset{(11)}{\Rightarrow} caCBAB \overset{(13)}{\Rightarrow} cacBAB \overset{(12)}{\Rightarrow} cacbAB \overset{(11)}{\Rightarrow} cacbaB \overset{(12)}{\Rightarrow} cacbab$$