Contents

PART A

SEQUENTIAL GRAMMARS

Chapter 2

Basic Families of Grammars and Languages

2.1 Definitions and Examples

Any natural languages is based on some grammar, which contains the rules by which syntactically correct sentences of the language can be built. For instance, a sentence can consist of a noun phrase (subject) and a verb phrase (predicate) or of a noun phrase (subject), verb phrase (predicate) and a noun phrase (object); a noun phrase can be built by a determiner and a noun; a verb phrase can consist of a verb and an adverb. In a formalized way, we can describe these rules by

(a)	(sentence) → (noun phrase) (verb phrase),
(b)	(sentence) → (noun phrase) (verb phrase) (noun phrase),
(c)	(noun phrase) → (determiner) (noun),
(d)	(verb prase) → (verb) (adverb).

Moreover, the parts of speech (noun, verb, etc.) have to be replaced by words of this type. Thus we also have rules of the following forms.

(e1)	(noun) → dog,	(e2)	(noun) → banana,
(f1)	(determiner) → the,	(f2)	(determiner) → a,
(g1)	(verb) → goes,	(g2)	(verb) → sings,
(h)	(adverb) → slowly.		

By a successive application of such rules, the correct sentences of a natural language are constructed. For instance, by an application of (a), (d), (g1), (h), (c), (e1), and (f1) in this order, we get

$$\begin{aligned}
\text{(sentence)} &\Longrightarrow \text{(noun phrase) (verb phrase)}\\
&\Longrightarrow \text{(noun phrase) (verb) (adverb)}\\
&\Longrightarrow \text{(noun phrase) goes (adverb)}\\
&\Longrightarrow \text{(noun phrase) goes slowly}\\
&\Longrightarrow \text{(determiner) (noun) goes slowly}\\
&\Longrightarrow \text{(determiner) dog goes slowly}\\
&\Longrightarrow \text{the dog goes slowly}
\end{aligned}$$

which is a sentence of the English languages. However, we mention that that the rules cover only the syntax of a languages and not the semantics. Thus we are able to obtain

$$(\text{sentence}) \Longrightarrow \ldots \Longrightarrow \text{a banana sings slowly}$$

by the use of (a), (d), (g2), (h), (c), (e2), and (f2), which is a syntactically correct sentence but semantical nonsense.

The situation is analogous with respect to programming languages. Here the manual of a programming languages contains the rules by which programs, subprograms or parts of programs can be built. As an example we give some rules from PASCAL which describe real numbers (where our notation is analogous to that used above for natural languages and differs from the usual notation for programming languages; the symbol | divides existing possibilities for replacements; {X} denotes an arbitrary non-empty sequence of elements which can be taken for X).

(unsigned real) → (unsigned integer).(digit){digit} | (unsigned integer)E(scale factor)
(unsigned integer) → (digit) | (digit){digit}
(scale factor) → (unsigned integer) | (sign) (unsigned integer)
(digit) → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
(sign) → + | –

By applications of these rules we get

$$\begin{aligned}(\text{unsigned real}) &\Longrightarrow (\text{unsigned integer})\text{E}(\text{scale factor})\\ &\Longrightarrow (\text{digit})\{\text{digit}\}\text{E}(\text{scale factor})\\ &\Longrightarrow 3\{\text{digit}\}\text{E}(\text{scale factor})\\ &\Longrightarrow 314\text{E}(\text{scale factor})\\ &\Longrightarrow 314\text{E}(\text{sign})(\text{unsigned integer})\\ &\Longrightarrow 314\text{E}\text{–}(\text{unsigned integer})\\ &\Longrightarrow 314\text{E}\text{–}(\text{digit})\\ &\Longrightarrow 314\text{E}\text{–}2\end{aligned}$$

which gives the real number 3.14 (the well-known approximation for π).

We notice that the definition of natural and programming languages have the following in common:

- Essentially, the given rules describe a replacement. The objects given at the left hand side are replaced by the right hand side.
- There are some objects which are replaced (e. g. (noun phrase), (verb), (unsigned real), (digit)), and there are objects which are not changed by the replacements, i. e., they are terminal characters (e. g. the words of natural language or the digits 0,1,. . . ,9 and the signs + and –.
- The generation of sentences and programs starts with fixed objects (sentence) and (program) and ends if only unreplaceable objects are existing.

We now give a formalization of the intuitive ideas used in the description and/or generation of natural and programming languages from their grammars. We shall use letters of an alphabet as objects. Then the generated sentences or programs or subprograms are words over an alphabet which contains as letters words of the English language or basic commands of the programming language as `if`, `while` and digits, respectively.

In order to restrict the possibilities to choose the basic alphabet, in the sequel we assume that all considered alphabets are finite subsets of a fixed infinite set of countable cardinality.

Definition 2.1 *A* phrase structure grammar *(or short grammar) is a quadruple*

$$G = (N, T, P, S)$$

where

– *N and T are finite, disjoint alphabets, whose union is denoted by V,*
– *P is a finite subset of $(V^* \setminus T^*) \times V^*$, and*
– *S is an element of N.*

The elements of N are called *nonterminals* (or variables). ⟨noun phrase⟩ and ⟨unsigned real⟩ are typical examples of nonterminals if we consider natural or programming languages. The elements of T are *terminals.* In the sequel we mostly use capital Latin letters to denote nonterminal and small Latin letters for the notation of terminals; in both cases we use indexed version of such letters, too. The elements of P are called *rules* or *productions.* Usually, we write a pair (α, β) of P as $\alpha \to \beta$, since this notation reflects the idea that application of rules are replacements (see next definition). The distinguished symbol $S \in N$ is called the *axiom* or start word of the grammar.

Definition 2.2 *Let $G = (N, T, P, S)$ be a phrase structure grammar as in Definition 2.1. We say that the word $\gamma \in V^+$* directly derives *the word $\gamma' \in V^*$ (written as $\gamma \Longrightarrow \gamma'$), if*

$$\gamma = \gamma_1\alpha\gamma_2,\ \gamma' = \gamma_1\beta\gamma_2,\ \alpha \longrightarrow \beta \in P$$

for some $\gamma_1, \gamma_2 \in V^*$.

According to Definition 2.2 γ' is obtained from γ if the subword of α of γ is replaced by β, where the rule $\alpha \to \beta$ exists in P. Therefore the productions of P state which replacements are allowed. We also say that γ' is directly generated from γ. The transformation from γ to γ' is called a direct derivation step, too. If we want to emphasize the rule $p = \alpha \to \beta$ which is applied in the derivation step, we write

$$\gamma \underset{p}{\Longrightarrow} \gamma'.$$

By $\Longrightarrow$ a binary relation on V^* is defined. As usual we can build the reflexive and transitive closure $\overset{*}{\Longrightarrow}$ of $\Longrightarrow$, i. e.,

$$\gamma \overset{*}{\Longrightarrow} \gamma'$$

if and only if there are a natural number $n \geq 0$ and words $\delta_0,\ \delta_1,\ \delta_2, \ldots,\ \delta_{n-1},\ \delta_n$ such that

$$\gamma = \delta_0 \Longrightarrow \delta_1 \Longrightarrow \delta_2 \Longrightarrow \ldots \Longrightarrow \delta_{n-1} \Longrightarrow \delta_n = \gamma'$$

(in case $n = 0$ we have $\gamma' = \gamma$, and in case $n = 1$ we have $\gamma \Longrightarrow \gamma'$). Thus $\gamma \overset{*}{\Longrightarrow} \gamma'$ if and only if γ' is obtained from γ by iterated applications of (not necessarily identical) rules of P. If $\gamma \overset{*}{\Longrightarrow} \gamma'$, then we say that γ derives or generates γ' (in some derivation steps). If we want to mention which production are used to get a derivation $\gamma \overset{*}{\Longrightarrow} \gamma'$, we write $\gamma \underset{q}{\Longrightarrow} \gamma'$ where q is the sequence of productions used in the derivation.

If a derivation $\gamma \overset{*}{\Longrightarrow} \gamma'$ consists of n direct derivation steps, then we say that the derivation has the *length* n and write sometimes $\gamma \overset{n}{\Longrightarrow} \gamma'$.

A word $w \in V^*$ is called *sentential form* of G, if $S \overset{*}{\Longrightarrow} w$, i. e., if w can be obtained by applications of rules of P from the axiom S.

Definition 2.3 *For a phrase structure grammar $G = (N, T, P, S)$ as in Definition 2.1, we define the* language $L(G)$ generated by G *by*

$$L(G) = \{w \mid w \in T^* \text{ and } S \overset{*}{\Longrightarrow} w\}.$$

According to this definition the language generated by $G = (N, T, P, S)$ consists of all sentential forms of G which only contain terminal letters. Moreover, this definition shows that it is necessary to give the letter S as a component in the grammar because only those words belong to the language which can be generated from S. Furthermore, by this definition it is clear why the elements of N and T are called nonterminals and terminals, respectively; the nonterminals do not occur in the words of the generated language, but they are necessary to perform the derivations; the terminals have a final character since they cannot be replaced and form the alphabet for the generated language.

We present some examples.

Example 2.4 We consider the phrase structure grammar

$$G_1 = (\{S, A, B\},\ \{a, b\},\ \{p_1, p_2, p_3, p_4, p_5\},\ S)$$

with

$$p_1 = S \to AB,\ p_2 = A \to aA,\ p_3 = A \to \lambda,\ p_4 = B \to Bb,\ p_5 = B \to \lambda.$$

We first prove that any sentential form of G_1 has one of the following forms:

$$S,\ a^nABb^m,\ a^nAb^m,\ a^nBb^m,\ a^nb^m \text{ with } n \geq 0 \text{ and } m \geq 0. \tag{2.1}$$

Obviously, the statement holds for the axiom S and the only word AB ($n = m = 0$) which can directly be derived from S. We consider a word of the form a^nABb^m. Then we have the following derivations:

$$\begin{array}{ll} a^nABb^m \underset{p_2}{\Longrightarrow} a^naABb^m, & a^nABb^m \underset{p_3}{\Longrightarrow} a^n\lambda Bb^m, \\ a^nABb^m \underset{p_4}{\Longrightarrow} a^nABbb^m, & a^nABb^m \underset{p_5}{\Longrightarrow} a^nA\lambda b^m. \end{array}$$

Thus, from a^nABb^m, we obtain only the words

$$a^{n+1}ABb^m,\ a^nBb^m,\ a^nABb^{m+1},\ a^nAb^m$$

which all have the form given in (2.1). Analogously, it is easy to prove that all words derivable from a^nAb^m and a^nBb^m have the form given in (2.1). Since we cannot apply a rule to a word of the form a^nb^m, we have proved the above statement.

We now show that all words of the forms given in (2.1) are sentential forms of G_1. With exception of a^nAb^m this assertion follows from the derivation

$$\begin{aligned} S &\underset{p_1}{\Longrightarrow} \underbrace{AB \Longrightarrow aAB \Longrightarrow aaAB \Longrightarrow \ldots \Longrightarrow a^{n-1}AB}_{(n-1)\ applications\ of\ p_2} \\ &\underset{p_2}{\Longrightarrow} \underbrace{a^nAB \Longrightarrow a^nABb \Longrightarrow a^nABb^2 \Longrightarrow \ldots \Longrightarrow a^nAb^m}_{m\ applications\ of\ p_4} \\ &\underset{p_3}{\Longrightarrow} a^nBb^m \underset{p_5}{\Longrightarrow} a^nb^m. \end{aligned}$$

Moreover, a^nAb^m is obtained by an analogous derivation where we change the order of applications of p_3 and p_5.

Since the language generated by G_1 only contains words over the terminal alphabet $\{a, b\}$, it consists of all words of $\{a, b\}^*$ which have a form given in (2.1). Consequently,

$$L(G_1) = \{a^nb^m \mid n \geq 0,\ m \geq 0\}.$$

Example 2.5 Let

$$G_2 = (\{S\},\ \{a, b\},\ \{S \to aSb, S \to ab\},\ S).$$

By induction we now show that a word w can be derived by n derivation steps, $n \geq 1$ if and only if $w = a^nSb^n$ or $w = a^nb^n$.

Obviously, the statement holds for $n = 1$ because we can generate only aSb and ab by an application of $S \to aSb$ and $S \to ab$, respectively.

Let us assume that w is obtained by n derivation steps. By definition, there is a word v which can be derived by $n - 1$ steps such that $v \Longrightarrow w$ by an application of a rule. By induction hypothesis, $v = a^{n-1}Sb^{n-1}$ or $v = a^{n-1}b^{n-1}$. Since we cannot apply a rule to $a^{n-1}b^{n-1}$, we get $v = a^{n-1}Sb^{n-1}$. By application of the two rules of the grammar to v we derive $a^{n-1}aSbb^{n-1} = a^nSb^n$ and $a^{n-1}abb^{n-1} = a^nb^n$. Thus our assertion is shown.

Since any word of $L(G_2)$ contains only terminal letters and can be generated by a certain number of derivation steps, we obtain

$$L(G_2) = \{a^nb^n \mid n \geq 1\}.$$

Example 2.6 We consider the phrase structure grammar

$$G_3 = (\{S, A\},\ \{a, b\},\ \{S \to \lambda, S \to aS, S \to Sb\},\ S).$$

As in Example 2.4 we show that the set of sentential form consists of all words of the form a^nSb^m or a^nb^m with $n \geq 1$ and $m \geq 1$ (or we prove as in Example 2.5 that by k derivation steps, $k \geq 1$, we can derive exactly the words a^nSb^m, $a^{n-1}b^m$, a^nb^{m-1} with $n + m = k$). Then

$$L(G_3) = \{a^nb^m \mid n \geq 0,\ m \geq 0\}$$

follows immediately.

Example 2.7 Let the phrase structure grammar

$$G_4 = (\{S, A\},\ \{a, b\},\ \{S \to \lambda, S \to aS, S \to a, S \to A, A \to bA, A \to b\},\ S)$$

be given. Essentially, in Figure 2.1 all possible derivation in the grammar G_4 are given, where the arrows to the right correspond to an application of $S \rightarrow aS$ or $A \rightarrow bA$, the downwards arrows correspond to applications of $S \rightarrow A$ or $A \rightarrow b$, and the upwards and leftupwards arrows correspond to applications of $S \rightarrow a$ and $S \rightarrow \lambda$, respectively. Now it is easy to see that we get

$$L(G_4) = \{a^n b^m \mid n \geq 0,\ m \geq 0\}.$$

A formal proof as in the preceding examples is left to the reader.

Example 2.8 We consider the phrase structure grammar

$$G_5 = (\{S, A, B, B', B''\},\ \{a, b, c\},\ \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8\},\ S)$$

with

$$\begin{array}{llll} p_1 = S \rightarrow ABA, & p_2 = AB \rightarrow aAbB', & p_3 = AB \rightarrow abB'', & p_4 = B'b \rightarrow bB', \\ p_5 = B''b \rightarrow bB'', & p_6 = B'A \rightarrow BAc, & p_7 = B''A \rightarrow c, & p_8 = bB \rightarrow Bb. \end{array}$$

We determine $L(G_5)$ by an analysis of all possible derivation.

For $n \geq 0$, let $w_n = a^n ABb^n Ac^n$.

We first discuss the case $n \geq 2$. There are only the rules p_2 and p_3 which can be applied to w_n.

Case 1: Application of p_2. We obtain the word $a^{n+1}AbB'b^nAc^n$. Now only p_4 is applicable, and the application of this rule yields $a^{n+1}AbbB'b^{n-1}Ac^n$, i. e., we have moved B' to the right by one position. Again, only p_4 is applicable, which results in a further movement of B' to the right by one position. This situation remains until we have derived the word $a^{n+1}Ab^{n+1}B'Ac^n$. Now we can only apply p_6, which gives $a^{n+1}Ab^{n+1}BAc^{n+1}$. Now only p_8 can be applied which yields a move of B to the left by one position. Again, only this movement is possible until we obtain $a^{n+1}ABb^{n+1}Ac^{n+1} = w_{n+1}$.

Case 2: Application of p_3. We obtain the word $a^{n+1}bB''b^nAc^n$. Now only p_5 is applicable, i. e., we have to move B'' to the right by one position. This situation remains until we get $a^{n+1}b^{n+1}B''Ac^n$. We can only apply p_7 and obtain the terminal word $a^{n+1}b^{n+1}c^{n+1}$.

Analogously, we have only the following derivations starting from w_0 and w_1:

$$w_0 \overset{*}{\Longrightarrow} w_1, \quad w_0 \overset{*}{\Longrightarrow} abc, \quad w_1 \overset{*}{\Longrightarrow} w_2, \quad w_1 \overset{*}{\Longrightarrow} a^2b^2c^2.$$

Because $S \Longrightarrow w_0$ is the only derivation from S, we get

$$L(G_5) = \{a^n b^n c^n \mid n \geq 1\}.$$

Example 2.9 Let the phrase structure grammar

$$G_6 = (\{S, A, B, B', B''\},\ \{a, b, c\},\ \{p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8\},\ S)$$

with

$$\begin{array}{lll} p_0 = S \rightarrow abc, & p_1 = S \longrightarrow aABbA, & p_2 = AB \longrightarrow aAbB', \\ p_3 = AB \longrightarrow abB'', & p_4 = B'b \longrightarrow bB', & p_5 = B''b \longrightarrow bB'', \\ p_6 = B'A \longrightarrow BAc, & p_7 = B''A \longrightarrow cc, & p_8 = bB \longrightarrow Bb \end{array}$$

$$\begin{tikzcd}
\lambda & a & aa & aaa & aaaa & a^5 & a^6 & \cdots \\
 & S \arrow[r, Rightarrow] \arrow[u, Rightarrow] \arrow[ul, Rightarrow] \arrow[dddddddddddd, Rightarrow] & aS \arrow[r, Rightarrow] \arrow[u, Rightarrow] \arrow[ul, Rightarrow] \arrow[dddddddddd, Rightarrow] & aaS \arrow[r, Rightarrow] \arrow[u, Rightarrow] \arrow[ul, Rightarrow] \arrow[dddddddd, Rightarrow] & aaaS \arrow[r, Rightarrow] \arrow[u, Rightarrow] \arrow[ul, Rightarrow] \arrow[dddddd, Rightarrow] & a^4S \arrow[r, Rightarrow] \arrow[u, Rightarrow] \arrow[ul, Rightarrow] \arrow[dddd, Rightarrow] & a^5S \arrow[r, Rightarrow] \arrow[u, Rightarrow] \arrow[ul, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & & & & & & a^5A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & & & & & & a^5b & \cdots \\
 & & & & & a^4A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & a^4bA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & & & & & a^4b & a^4bb & \cdots \\
 & & & & a^3A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & a^3bA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & a^3b^2A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & & & & a^3b & a^3b^2 & a^3b^3 & \cdots \\
 & & & aaA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & aabA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & aabbA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & aab^3A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & & & aab & aabb & aab^3 & aab^4 & \cdots \\
 & & aA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & abA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & ab^2A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & ab^3A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & ab^4A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & & ab & ab^2 & ab^3 & ab^4 & ab^5 & \cdots \\
 & A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & bA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & bbA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & bbbA \arrow[r, Rightarrow] \arrow[d, Rightarrow] & b^4A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & b^5A \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \cdots \\
 & b & bb & bbb & b^4 & b^5 & b^6 & \cdots
\end{tikzcd}$$

Figure 2.1: Derivations in Example 2.7

be given. As in the preceding example, we can show that

$$L(G_6) = \{a^n b^n c^n \mid n \geq 1\}.$$

Example 2.10 We regard the phrase structure grammar

$$G_7 = (\{S\},\ \{x, y, z, +, -, \cdot, :, (,)\},\ P,\ S)$$

with

$$P = \{S \longrightarrow (S{+}S), S \longrightarrow (S{-}S), S \longrightarrow (S{\cdot}S), S \longrightarrow (S : S), S \longrightarrow x, S \longrightarrow y, S \longrightarrow z\}.$$

We show that $L(G_7)$ consists of all correctly bracketed arithmetic expression over the variables x, y, z (where we do not use precedence rules for the operations, which means that no brackets can be omitted, and do also not omit the most outer brackets).

As a first step we prove that any sentential form of G_7 is a correctly bracketed expression over the variables S, x, y, z. This follows easily by induction on the number of derivation steps since the axiom is a correct arithmetic expression and any replacement of S in a correctly bracketed expression by x, y, z or $(S \circ S)$, $\circ \in \{+, -, \cdot, :\}$ gives a correctly bracketed expression, again. Thus the languages $L(G_7)$ can only contain correctly bracketed arithmetic expressions over $\{x, y, z\}$.

We now show that all correctly bracketed expressions can be generated by G_7. We give a proof by induction on the number n of steps in the construction of arithmetic expressions. By no step of construction ($n = 0$), we can only obtain the basic arithmetic expressions, i. e., the variables x, y, z. Obviously, these can be generated by application of the rules $S \rightarrow x$, $S \rightarrow y$ and $S \rightarrow z$. Thus $x, y, z \in L(G_7)$. Let $n \geq 1$, and let w be an arithmetic expression which can be obtained in n steps of construction. Then $w = (w_1 \circ w_2)$ where $\circ \in \{+, -, \cdot, :\}$ and w_1 and w_2 are arithmetic expressions. Moreover, w_1 and w_2 can be constructed by at most $n - 1$ steps. By induction hypothesis, there are derivations

$$S \overset{*}{\Longrightarrow} w_1 \quad \text{and} \quad S \overset{*}{\Longrightarrow} w_2.$$

Thus we also have the derivation

$$S \Longrightarrow (S \circ S) \overset{*}{\Longrightarrow} (w_1 \circ S) \overset{*}{\Longrightarrow} (w_1 \circ w_2) = w.$$

Therefore $w \in L(G_7)$ is shown.

We now introduce some special types of grammars.

Definition 2.11 *A grammar $G = (N, T, P, S)$ is called*

i) monotone, *if $|\alpha| \leq |\beta|$ holds for all rules $\alpha \rightarrow \beta$ of P with the possible exception of $S \rightarrow \lambda$ if S does not occur on the right side of any production of P.*

ii) context-sensitive, *if all rules of P are of the form $uAv \rightarrow uwv$ with $u \in V^*$, $v \in V^*$, $A \in N$, and $w \in V^+$ with the possible exception of $S \rightarrow \lambda$ if S does not occur on the right side of any production of P.*

iii) context-free, *if all rules of P are of the form $A \rightarrow w$ with $A \in N$ and $w \in V^*$.*

iv) linear, *if all rules of P are of the form $A \rightarrow uBv$ or $A \rightarrow w$ with $A, B \in N$ and $u, v, w \in T^*$.*

v) regular, *if all rules of P are of the form $A \rightarrow wB$ or $A \rightarrow w$ with $A, B \in N$ and $w \in T^*$.*

The monotone grammars have – apart from the exceptional case – the property that, for any derivation step $\gamma \Longrightarrow \gamma'$, the length of the derived word γ' is not smaller than the length of γ, i. e., the derivation relation $\Longrightarrow$ is monotone with respect to the length. Therefore these grammars are called monotone.

By the application of a rule $uAv \rightarrow uwv$ of a context-sensitive grammar, we only replace the nonterminal A by the word w, however, this replacement is only allowed if u and v are left and right of A, respectively, in the sentential form. Thus the replacement requires the context u to the left and v to the right of A. This justifies the name context-sensitive.

In a context-free grammar, the application of the rule $A \rightarrow w$ also consists in the replacement of A by w, however, this can be done independent of the context of A. Note that A is not free of context, but the replacement does not depend of the context.

Linear and regular grammars are special cases of context-free grammars since the right hand side of a rule has a restricted form. In a linear (and regular) grammar, any sentential form contains at most one nonterminal. In a regular grammar, this nonterminal is the last letter of the sentential form.

Since the empty word can occur on the right hand side of a production of a context-free or linear or regular grammar, it is obvious that the empty word can be generated by grammars of these types. The exceptional case in the definition of monotone and context-sensitive grammars ensures that we can also generate the empty word in such grammars.

Essentially, the classification of grammars given in Definition 2.11 was introduced by NOAM CHOMSKY[1], but he used the terms type-0, type-1, type-2 and type-3 grammars for arbitrary, context-sensitive, context-free and regular grammars, respectively.

Let us determine the type of the grammars given in the examples 2.4 – 2.10 above.

The grammar G_1 is not monotone and not context-sensitive since its set of rules contains the rule $p_3 = A \rightarrow \lambda$. It is not linear and not regular by the rule $p_1 = S \rightarrow AB$. Obviously, G_1 is context-free.

The grammar G_2 is monotone, context-sensitive (one has to take $u = v = \lambda$ for all rules), context-free and linear, but it is not regular by the rule $S \rightarrow aSb$.

The grammar G_3 is not monotone and not context-sensitive because it has the rules $S \rightarrow \lambda$ and $S \rightarrow aS$. It is context-free and linear. But G_3 is not a regular grammar by the rule $S \rightarrow Sb$.

The grammar G_4 is regular, linear and context-free, but it is not monotone and not context-sensitive.

The grammar G_5 has none of the properties given in Definition 2.11. The grammar G_6 is monotone, but it is neither context-sensitive nor context-free nor linear nor regular.

Finally, the grammar G_7 is context-free, context-sensitive and monotone, however, it is not a linear grammar and not a regular grammar.

From Definition 2.11, it follows immediately that

- any context-sensitive grammar is monotone (because $|uAv| \leq |uwv|$ since w is not the empty word),
- any regular grammar is linear grammar,
- any linear grammar is a context-free grammar.

Definition 2.12 *A language L is called a monotone (context-sensitive, context-free, linear, and regular, respectively), if there exists a monotone (context-sensitive, context-free, linear, and regular, respectively) grammar G such that $L = L(G)$ holds.*

[1]linguist, computer scientist and philosopher, born in 1928 in Philadelphia (U.S.A.)

According to Definition 2.12, the language $L = \{a^n b^n \mid n \geq 1\}$ is a linear language because, by Example 2.5, $L = L(G_2)$ and G_2 is a linear grammar.

The language $L' = \{a^n b^m \mid n \geq 0,\ m \geq 0\}$ is a context-free language since, by Example 2.6, $L' = L(G_3)$ and G_3 is a context-free grammar. Although G_3 is not a regular grammar, we cannot say that L' is not a regular language; the grammar G_4 is regular and generates L' (see Example 2.7), i. e., L' is a regular language, too.

Analogously, $L'' = \{a^n b^n c^n \mid n \geq 0$ is generated by the non-monotone grammar G_5 (see Example 2.8), but nevertheless L'' is a monotone language since it is also generated by the monotone grammar G_6 (see Example 2.9).

By $\mathcal{L}(MON)$, $\mathcal{L}(CS)$, $\mathcal{L}(CF)$, $\mathcal{L}(LIN)$ and $\mathcal{L}(REG)$ we denote the families of all monotone, context-sensitive, context-free, linear, and regular languages, respectively. The family of all languages which can be generated by (arbitrary) phrase structure grammars is designated by $\mathcal{L}(RE)$. The languages in $\mathcal{L}(RE)$ are called recursively enumerable.[2]

Lemma 2.13 *Let X and Y be two types of grammars. If any grammar of type X is also a grammar of type Y, then $\mathcal{L}(X) \subseteq \mathcal{L}(Y)$.*

Proof. We have to prove that any language $L \in \mathcal{L}(X)$ belongs to $\mathcal{L}(Y)$.

Let $L \in \mathcal{L}(X)$. Then there is a grammar G of type X such that $L = L(G)$. By supposition, G is also a grammar of type Y and $L = L(G)$ is in $\mathcal{L}(Y)$, too. □

The following statement is a direct consequence of Lemma 2.13 and the remark above on the relation between the regular, linear, context-free, context-sensitive, and monotone grammars.

Lemma 2.14 *$\mathcal{L}(CS) \subseteq \mathcal{L}(MON) \subseteq \mathcal{L}(RE)$ and $\mathcal{L}(REG) \subseteq \mathcal{L}(LIN) \subseteq \mathcal{L}(CF) \subseteq \mathcal{L}(RE)$.*

In the following two sections we discuss, whether the inclusions of Lemma 2.14 are proper, and add a relation between the families of context-free and context-sensitive languages.

In order to describe the derivation of a word in a context-free grammar a tree is often associated with the derivation. The tree is called *derivation tree* of the derivation. The (inductive) construction of the tree is done in such a way that the root of the tree is the axiom and the leaves of the tree yield the sentential form if they are read from left to right. We describe the tree as a graph, i. e., as a pair (K, E) where K is the set of nodes and E is the set of edges.

Let a context-free grammar $G = (N, T, P, S)$ be given, and let $A \overset{*}{\Longrightarrow} w$ be a derivation of length n.

If $n = 0$, then no derivation step is done and we have $w = S$. The derivation tree is given as $(\{S\}, \emptyset)$, i. e., it is the graph which has only a single node S and no edge. Therefore S is root as well as leaf.

Let $n = 1$. Then the derivation has the form $S \Longrightarrow w$, where the rule $S \longrightarrow w$ is applied. Let $w = x_1 x_2 \dots x_m$ with $x_i \in N \cup T$, $1 \leq i \leq m$. Then we set $t = (K, E)$, where K consists of the symbols $S, x_1, x_2, \dots, x_m$ and E consists of the edges (S, x_i), $1 \leq i \leq m$. Thus S is the root and $x_1, x_2, \dots, x_m$ are the leaves of t. We arrange the edges in such way that reading the leaves from left to right gives the word $w = x_1 x_2 \dots x_m$.

[2]The justification for this term will be given in Subsection **??**.

Let $n \geq 2$. Then there is a derivation

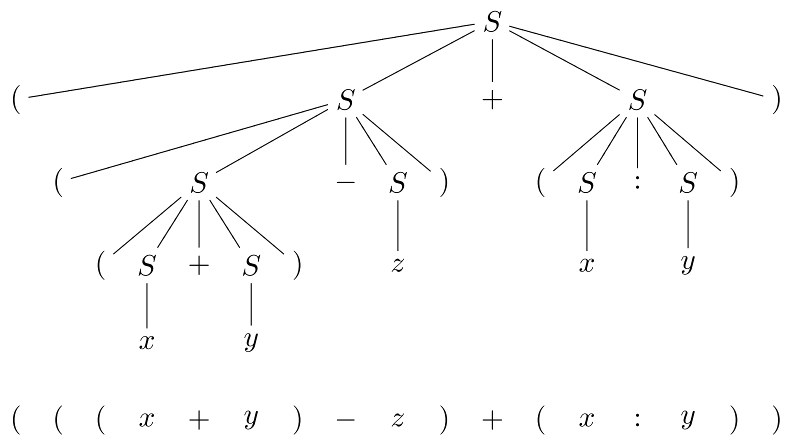
$$S \overset{*}{\Longrightarrow} u = y_1 y_2 \dots y_s A z_1 z_2 \dots z_r \Longrightarrow y_1 y_2 \dots y_s x_1 x_2 \dots x_m z_1 z_2 \dots z_r = w,$$

where $x_i, y_j, z_k \in N \cup T$ for $1 \leq i \leq m,\ 0 \leq j \leq s,\ 0 \leq k \leq r$ and the rule $A \to x_1 x_2 \dots x_m$ is applied in the last direct derivation step. The derivation $S \Longrightarrow^* u$ has the length $n-1$ and the derivation tree $t' = (K', E')$ associated with u has the root S and its leaves yield u if they are read from left to right We construct the derivation tree $t = (K, E)$ of w by the settings

$$K = K' \cup \{x_1, x_2, \dots, x_m\} \quad \text{and} \quad E = E' \cup \{(A, x_i) \mid 1 \leq i \leq m\},$$

where we arrange the new edges such that the reading of their leaves from left to right yields $x_1 x_2 \dots x_m$.

As an illustration we give the derivation tree of the word $(((x+y)-z)+(x:y))$, that is generated by the context-free grammar G_6 given in Example 2.10. Below the tree we write the leaves for illustrating that we obtain the sentential form under consideration if we read the leaves from left to right.



We note that a derivation tree for a sentential form of a monotone or context-sensitive grammar does not exist since the tree structure only allows that one nonterminal is at the left hand side of a production.

Finally, in this section, we introduce a measure $k(G)$ for the size of a phrase structure grammar $G = (N, T, P, S)$. Intuitively, we consider the (N, T, P, S) as a word over the alphabet consisting of the symbols of N and T and the technical symbols $(,), \{, \}, \to$ and the comma and take the length of the word as the size of the grammar. Obviously, we need one pair of brackets (parenthesis), three pairs of braces, three commas separating the four components of a grammar, $\#(N)$ symbols of N and $\#(N)-1$ commas between the symbols for the description of N, similarly, $2\#(T)-1$ letters for the description of T, $\#(P)$ times the arrow $\to$, $\#(P)-1$ commas separating the rules, $\sum_{\alpha\to\beta\in P} |\alpha\beta|$ symbols to write down the right hand and left hand sides of the rules (note that the empty word on a left hand side of a rule is not counted), and finally one letter for the axiom. Therefore,

$$k(G) = 9 + 2\#(N) + 2\#(T) + 2\#(P) + \sum_{\alpha\to\beta\in P} |\alpha\beta|.$$

For example, we have

$$k(G_1) = 40, \ k(G_2) = 26, \ k(G_3) = 30, \ \text{and } k(G_5) = 76.$$

2.2 Normal forms

In this section we show that, for the grammars of the types introduced in the preceding section, there exist normal forms, i. e., grammars of this type with further restrictions to the rules. The term normal form is justified by the following property of grammars in normal form. Any language which can be generated by grammars of a certain type can also be generated by grammars of this type in normal form. In addition to the existence, we show that the normal forms can effectively constructed from given grammars.

In the computations of the running time of such an construction we assume that that the following problems can be solved in one step: deciding whether an element belongs to a certain set, adding an new element to a set, deciding whether a symbol occurs in a word, deciding whether a rule has a certain form. Thus we are independent of the data structure which is used for the representation of the set, the word etc. If somebody wants to have more precise complexities, it is necessary to use the precise time to perform the above mentioned problems according to the used data structure. However, we note that the precise values are at most linear in the size of the grammar.

In the sequel we shall often use the normal forms as a technical tool (in proofs we can restrict to grammars in normal form). In this section, by the normal forms, we add further containments of the language families introduced in the preceding section.

Lemma 2.15 *For any phrase structure grammar $G = (N, T, P, S)$, a grammar $G' = (N', T, P', S)$ can be constructed in time $O(k(G))$ such that*

- *all rules of P' have the form $\alpha \to \beta$ with $\alpha \in (N')^+$ and $\beta \in (N')^*$ or $A \to a$ with $A \in N'$ and $a \in T$,*
- *$L(G') = L(G)$, and*
- *$k(G') \in O(k(G))$.*

Moreover, if G is a monotone or context-sensitive or context-free grammar, then G' also is monotone or context-sensitive or context-free, respectively.

Proof. For any terminal a, let a' be a new symbol which is not in $N \cup T$ such that $a' \neq b'$ for different terminals a and b. We set

$$N' = N \cup \{a' \mid a \in T\}.$$

If $w = x_1x_2 \dots x_n$ is a word of V^*, $x_i \in V$ for $1 \leq i \leq n$, then we set $w' = y_1y_2 \dots y_n$ where

$$y_i = \begin{cases} x_i & \text{for } x_i \in N \\ x_i' & \text{for } x_i \in T \end{cases}$$

for $1 \leq i \leq n$. We construct the grammar $G' = (N', T, P', S)$ with

$$P' = \{\alpha' \longrightarrow \beta' : \alpha \longrightarrow \beta \in P\} \cup \{a' \longrightarrow a : a \in T\}.$$

Since we add $\#(T)$ new nonterminals and $\#(T)$ new rules (each of them consisting of four symbols (including the separating comma), it is obvious that $k(G') \leq 5 \cdot k(G)$.

Obviously, the adding of nonterminals and rules requires less than $4 \cdot k(G)$ steps. Moreover, the change of the rules of G to those of G' requires only the change from a to a' for all occurrences of all $a \in T$. Thus it can be done in at most $2 \cdot k(G)$ steps. Hence the construction of G' can be done in a time $O(k(G))$ and $k(G') \in O(k(G))$ holds.

We now prove that $L(G') = L(G)$.

Let $w \in L(G)$. Then there is a derivation

$$S = w_0 \Longrightarrow w_1 \Longrightarrow w_2 \Longrightarrow \ldots \Longrightarrow w_n = w$$

in G. According to the construction of P', there is a derivation

$$S = w'_0 \Longrightarrow w'_1 \Longrightarrow w'_2 \Longrightarrow \ldots w'_n = w' = v_0 \Longrightarrow v_1 \Longrightarrow v_2 \Longrightarrow \ldots \Longrightarrow v_m = w$$

in G' where, for $0 \leq i \leq n-1$, we use the production $\alpha' \to \beta'$ in the derivation step $w'_i \Longrightarrow w'_{i+1}$, if $\alpha \to \beta$ is applied in the derivation step $w_i \Longrightarrow w_{i+1}$ and, for $0 \leq j \leq m-1$, we apply rules of the form $a' \to a$ in the derivation steps $v_j \Longrightarrow v_{j+1}$, i. e., we replace all primed letters in w' by their unprimed version. Thus we get $w \in L(G')$ which proves $L(G) \subseteq L(G')$.

Let now $x \in L(G')$. Then there is derivation $S \overset{*}{\Longrightarrow} x$ in G'. We change the order of the application of rules in this derivation in such a way that we first apply only rules of the form $\alpha' \to \beta'$ and finally only rules of the form $a' \to a$, i. e., we postpone the applications of rules $a' \to a$ until we have already applied all rules $\alpha' \to \beta'$. Since after an application of the form $a' \to a$, the generated a cannot be involved in the application of other rules of P', this change of the order does not change the language. Therefore there is a derivation

$$S = x'_0 \Longrightarrow x'_1 \Longrightarrow x'_2 \Longrightarrow \ldots \Longrightarrow x'_r = x' = y_0 \Longrightarrow y_1 \Longrightarrow y_2 \Longrightarrow \ldots \Longrightarrow y_s = x$$

in G' where, for with $x_i \in (N')^*$ where, for $0 \leq i \leq r-1$, $x'_{i+1} \in (N')^*$ and the derivation step $x'_i \to x'_{i+1}$ is obtained by an application of a rule of the form $\alpha' \to \beta'$, and for $1 \leq j \leq s$, we apply a rule of the form $a' \to a$ to get $y_j \to y_{j+1}$. Then we have the derivation

$$S = x_0 \Longrightarrow x_1 \Longrightarrow x_2 \Longrightarrow \ldots \Longrightarrow x_n = x$$

in G. Therefore $L(G') \subseteq L(G)$.

From the shown inclusions, we get $L(G') = L(G)$.

By the construction a rule $\alpha \to \beta$ of P with $|\alpha| \leq |\beta|$ is transformed into a production $\alpha' \longrightarrow \beta'$ with $|\alpha'| \leq |\beta'|$ because $|\alpha| = |\alpha'|$ and $|\beta| = |\beta'|$. Thus monotonicity is preserved by the construction. Furthermore, a rule $uAv \to uwv$ is transformed into $u'Av' \to u'w'v'$. Therefore context-sensitivity and context-freeness are also preserved. □

We want to illustrate our constructions by an example. For this purpose, we consider the phrase structure grammar

$$G_8 = (\{S, A, B, C\}, \{a, b\}, \{S \to AB,\ AB \to CAB,\ CA \to aC,\ CB \to b\}, S\,).$$

It is easy to see that $L(G_8) = \{a^n b \mid n \geq 1\}$. Following the construction in the proof of Lemma 2.15 we get the grammar

$$G_8^{(1)} = (\,\{S, A, B, C, a', b'\},\ \{a,\ b\}, P_8',\ S\,)$$

with

$$P_8^{(1)} = \{S \to AB,\ AB \to CAB,\ CA \to a'C,\ CB \to b',\ a' \to a,\ b' \to b\}.$$

We now give a result which tells us that the essential difference between arbitrary phrase structure grammars and monotone grammars is given by the existence of erasing rules $X \to lambda$ (where X is a nonterminal).

Lemma 2.16 *For any phrase structure grammar $G = (N, T, P, S)$, a grammar $G = (N', T, P', S)$ can be constructed in a time $O(k(G))$ such that*
- *any rule of P' has the form $\alpha \to \beta$ with $\alpha \in (N')^+$, $\beta \in (N')^+$ and $|\alpha| \leq |\beta|$ or $A \to a$ or $A \to \lambda$ with $A \in N'$ and $a \in T$ and*
- *$L(G') = L(G)$, and*
- *$k(G') \in O(k(G))$.*

Proof. Let $G = (N, T, P, S)$ be a phrase structure grammar. Then we first construct the phrase structure grammar $G'' = (N'', T, P'', S)$ as in the proof of Lemma 2.15. We build the phrase structure grammar $G' = (N', T, P', S)$ where
- $N' = N'' \cup \{D\}$ where D is a new symbol, and
- P' is obtained from P'' by replacing any rule $\alpha \to \beta \in P''$ with $|\alpha| > |\beta|$ by the rule $\alpha \to \beta D^{|\alpha|-|\beta|}$ and adding the new rule $D \to \lambda$.

Obviously, all rules of P' have the required form. Thus it remains to show that $L(G') = L(G)$ holds. In order to see this, we mention that the change of the rule (from $\alpha \to \beta$ to $\alpha \to \beta D^{|\alpha|-|\beta|}$ does not change the generated language because the additional letters D are cancelled by applying $D \to \lambda$ and there are no other rules for D. Thus we have $L(G) \subseteq L(G')$. By an analogous argument the reverse change does also not change the language, i. e., $L(G') \subseteq L(G)$.

Since we only add one new nonterminal and one new rule consisting of two symbols and $|\alpha\beta D^{|\alpha|-|\beta|}| = 2|\alpha| \leq 2|\alpha\beta|$ for each new rule $\alpha \to \beta D^{|\alpha|-|\beta|}$, it is obvious, that $k(G') \leq 2k(G'')$. To perform the transformation from G'' to G', we need at most $2k(G'')$ steps. By Lemma 2.15, the construction of G' from G can be done in time $O(k(G))$ and $k(G') \in O(k(G))$. □

If we apply this construction to our grammar $G_8^{(1)}$ constructed above, then we get

$$G_8^{(2)} = (\,\{S, A, B, C, D, a', b', D\},\ \{a, b\},\ P_8'',\ S\,)$$

with

$$P_8^{(2)} = \{S \to AB,\ AB \to CAB,\ CA \to a'C,\ CB \to b'D,\ a' \to a,\ b' \to b,\ D \to \lambda\}.$$

We say that a phrase structure grammar $G = (N, T, P, S)$ is of *order* n, if $|\alpha| \leq n$ and $|\beta| \leq n$ for any rule $\alpha \to \beta$ of P. We show that we can restrict the order of grammars to two.

Lemma 2.17 *Let $n \geq 3$. For any phrase structure grammar $G = (N, T, P, S)$ of order n such that any rule of P has the form $\alpha \to \beta$ with $\alpha \in N^+$, $\beta \in N^+$ and $|\alpha| \leq |\beta|$ or $A \to a$ or $A \to \lambda$ with $A \in N$ and $a \in T$, a phrase structure grammar G' of order $n-1$ can be constructed such that $L(G') = L(G)$. Moreover, if G is monotone, then G' is a monotone grammar, too.*

Proof. Let $G = (N, T, P, S)$ be a phrase structure grammar of order n where any rule of P has the form $\alpha \to \beta$ with $\alpha \in N^+$, $\beta \in N^+$ and $|\alpha| \leq |\beta|$ or $A \to a$ or $A \to \lambda$ with $A \in N'$ and $a \in T$. We divide P into three subsets

$$\begin{aligned} P_1 &= \{\alpha \to \beta \mid \alpha \to \beta \in P,\ |\alpha| \leq 2,\ |\beta| \leq 2\}, \\ P_2 &= \{\alpha \to \beta \mid \alpha \to \beta \in P,\ |\alpha| \geq 2,\ |\beta| \geq 3\}, \\ P_3 &= \{\alpha \to \beta \mid \alpha \to \beta \in P,\ |\alpha| = 1,\ |\beta| \geq 3\}. \end{aligned}$$

Obviously, $P = P_1 \cup P_2 \cup P_3$.

We construct a phrase structure $G' = (N', T, P', S)$ as follows: If p is a rule in P_2, then it has the form $AB\alpha' \to CDE\beta'$ with $\alpha' \in N^*$ and $\beta' \in N^*$ and we set

$$N_p = \{A_p, B_p\} \text{ and } P_p = \{AB \to A_pB_p,\ A_p \to C,\ B_p\alpha' \to DE\beta'\}$$

where A_p und B_p are additional symbols (not occurring in N and satisfying $\{A_p, B_p\} \cap \{A_q, B_q\} = \emptyset$ for different rules p and q of P_2). If p is a rule in P_3, then it has the form $A \to CDE\beta'$ with $\beta' \in N^*$ and we put

$$N_p = \{B_p\} \text{ and } P_p = \{A \to CB_p,\ B_p \to DE\beta'\} \tag{2.2}$$

where A_p und B_p are additional symbols, again. We now set

$$N' = N \cup \bigcup_{p \in P_2 \cup P_3} N_p \quad \text{and} \quad P' = P_1 \cup \bigcup_{p \in P_2 \cup P_3} P_p\,.$$

It is easy to see that G' is of order $n-1$ and that monotonicity is preserved.

We first prove that $L(G) \subseteq L(G')$. This follows easily from the fact that any production of P_1 can be applied in both grammars, a derivation step $\gamma_1 AB\alpha'\gamma_2 \Longrightarrow \gamma_1 CDE\beta'\gamma_2$ according to an application of $p = AB\alpha \to CDE\beta' \in P_2$ in G can be simulated in G' by the three step derivation

$$\gamma_1 AB\alpha'\gamma_2 \Longrightarrow \gamma_1 A_pB_p\alpha'\gamma_2 \Longrightarrow \gamma_1 CB_p\alpha'\gamma_2 \Longrightarrow \gamma_1 CDE\beta\gamma_2$$

using the rules of P_p in the given order, and a derivation step $\gamma_1 A\gamma_2 \Longrightarrow \gamma_1 CDE\beta'\gamma_2$ using $p = A\alpha \to CDE\beta' \in P_3$ in G can be simulated in G' by the three step derivation

$$\gamma_1 A\alpha'\gamma_2 \Longrightarrow \gamma_1 A_pB_p\alpha'\gamma_2 \Longrightarrow \gamma_1 CB_p\alpha'\gamma_2 \Longrightarrow \gamma_1 CDE\beta\gamma_2.$$

Thus any derivation in G can be transformed into a derivation in G'.

In order to prove the converse inclusion we consider a derivation of $w \in L(G')$ in G' which contains an application of $AB \to A_pB_p$ for some rule $p = AB\alpha' \to CDE\beta' \in P_2$

(note that the other two rules of P_p can only be applied if $AB \to A_pB_P$ was already applied earlier in the derivation). This derivation has the form

$$\begin{aligned} S &\overset{*}{\Longrightarrow} \gamma_1 AB\alpha'\gamma_2 \Longrightarrow \gamma_1 A_pB_p\alpha'\gamma_2 \underset{q_1}{\Longrightarrow} \gamma_1' A_pB_p\alpha'\gamma_2' \\ &\Longrightarrow \gamma_1' CB_p\alpha'\gamma_2' \underset{q_2}{\Longrightarrow} \gamma_1'' B_p\alpha'\gamma_2'' \Longrightarrow \gamma_1'' DE\beta'\gamma_2'' \overset{*}{\Longrightarrow} w \in T^*, \end{aligned} \tag{2.3}$$

where
– q_1 is the sequence of productions applied after the application of $AB \to A_pB_p$ and before the application of $A_p \to C$ and which realize $\gamma_1 \overset{*}{\Longrightarrow} \gamma_1'$ and $\gamma_2 \overset{*}{\Longrightarrow} \gamma_2'$, and
– q_2 is the sequence of productions realizing $\gamma_1' C \overset{*}{\Longrightarrow} \gamma_1''$ and $\gamma_2' \overset{*}{\Longrightarrow} \gamma_2''$,
or

$$\begin{aligned} S &\overset{*}{\Longrightarrow} \gamma_1 AB\alpha'\gamma_2 \Longrightarrow \gamma_1 A_pB_p\alpha'\gamma_2 \underset{q_1'}{\Longrightarrow} \gamma_1' A_pB_p\alpha'\gamma_2' \\ &\Longrightarrow \gamma_1' A_pDE\beta'\alpha'\gamma_2' \underset{q_2'}{\Longrightarrow} \gamma_1'' A_p\gamma_2'' \Longrightarrow \gamma_1'' C\gamma_2'' \overset{*}{\Longrightarrow} w \in T^*, \end{aligned} \tag{2.4}$$

where
– q_1' is the sequence of productions realizing $\gamma_1 \overset{*}{\Longrightarrow} \gamma_1'$ and $\gamma_2 \overset{*}{\Longrightarrow} \gamma_2'$, and
– q_2 is the sequence of productions realizing $\gamma_1' \overset{*}{\Longrightarrow} \gamma_1''$ and $DE\beta'\gamma_2' \overset{*}{\Longrightarrow} \gamma_2''$.
For (2.3), there is a derivation

$$\begin{aligned} S &\overset{*}{\Longrightarrow} \gamma_1 AB\alpha'\gamma_2 \Longrightarrow \gamma_1 CDE\beta'\gamma_2 \underset{q_1}{\Longrightarrow} \gamma_1' CDE\beta'\gamma_2' \\ &\underset{q_2}{\Longrightarrow} \gamma_1'' DE\beta'\gamma_2'' \overset{*}{\Longrightarrow} w \in T^*, \end{aligned}$$

which – essentially – is obtained from (2.3) by replacing one application of the rules of P_p by an application of $p = AB\alpha' \to CDE\beta \in P$. Analogously, we can show that, in case of (2.4), we can also replace the application of P_p by p. Moreover, this also holds for the application of some P_q with $q \in P_3$. Thus, for $r \in P_2 \cup P_3$, in succession we can replace all applications of P_r by r, i.e., we obtain a derivation of w which only consists of applications of P. Thus $w \in L(G)$. This implies $L(G') \subseteq L(G)$. □

We continue our example. From $G_8^{(2)}$ we obtain

$$G_8^{(3)} = (\{A, B, C, D, a', b', D, , A', B'\}, \{a, b\}, P_8^{(3)}, S)$$

with

$$\begin{aligned} P_8^{(3)} = \{ & S \to AB,\ AB \to A_pB_p',\ A_p \to C,\ B_p' \to AB,\ CA \to a'C, \\ & CB \to b'D,\ a' \to a,\ b' \to b,\ D \to \lambda\} \end{aligned}$$

The grammars of order two have a special name.

Definition 2.18 *i) A phrase structure grammar $G = (N, T, P, S)$ is in Kuroda normal form*[3] *if all its productions have one of the following forms:*

$$A \to B,\ A \to BC,\ AB \to CD,\ A \to a,\ \text{or } A \to \lambda \quad \text{with} \quad A, B, C, D \in N \text{ and } a \in T.$$

[3]named after the japanese linguist SIGE-YUKI KURODA (1934–2009) who introduced the normal form in [18]

ii) A monotone grammar $G = (N, T, P, S)$ is called to be in Kuroda normal form if all its productions have one of the following forms:

$$A \to B,\ A \to BC,\ AB \to CD,\ \textit{or } A \to a \quad \textit{with} \quad A, B, C, D \in N \textit{ and } a \in T$$

with the possible exception of $S \to \lambda$ if S does not occur on the right side of any production of P.

Note that the grammar $G_8^{(3)}$ is a phrase structure grammar in Kuroda normal form.

We now prove that grammars in Kuroda normal form are sufficient to generate all languages.

Theorem 2.19 *For any phrase structure grammar G, there can be constructed a phrase structure grammar $\overline{G}$ in Kuroda normal form such that $L(\overline{G}) = L(G)$. Moreover, if G is monotone, then $\overline{G}$ is a monotone grammar, too.*

Proof. Let G be a phrase structure grammar. We construct in succession a grammar G' according to Lemma 2.15, from G' a grammar G'' according to Lemma 2.16. Then G'' satisfies the suppositions of Lemma 2.17.

Let G'' be of order n. If $n = 2$, then G'' is in Kuroda normal form and we choose $\overline{G} = G''$. If $n \geq 3$, we can construct G''' of order $n-1$ from G'' according to Lemma 2.17. It is easy to see that G''' satisfies the suppositions of Lemma 2.17, again. Therefore we can iterate the process until we get a grammar of order 2 which we take as $\overline{G}$.

If G is monotone, then we can omit the construction of G'' according to Lemma 2.16 since the properties obtained by that constructions are already satisfied by the monotone grammar G'. □

We now use the Kuroda normal form to prove that any monotone grammar can be converted into a context-sensitive grammar such that both grammars generate the same language.

Lemma 2.20 *For any monotone grammar G, there is a context-sensitive grammar G' such that $L(G') = L(G)$.*

Proof. Let G be a monotone grammar. By the lemmas given above, there is a monotone grammar G'' in Kuroda normal form such that $L(G'') = L(G)$. Let us assume that $P'' = P_1 \cup P_2$ where P_1 contains all rules of the form $A \to B$, $A \to BC$ and $A \to a$ and P_2 contains all rules of the form $AB \to CD$ (where $A, B, C, D \in N''$ and $a \in T$).

We now construct $G'' = (N'', T, P_1 \cup P_2', S)$ by the following procedure. For any rule $p = AB \to CD \in P_2$, we set

$$R_p = \{AB \to A_pB,\ A_pB \to A_pB_p',\ A_pB_p' \to CB_p',\ CB_p \to CD\},$$

where A_p and B_p are new nonterminal symbols (i. e., $A_p, B_p' \notin N''$, $A_p \neq B_p'$, $A_p \neq A_{p'}$, $A_p \neq B_{p'}'$, and $B_p' \neq B_{p'}'$ for all $p, p' \in P_2$, $p \neq p'$). We define P_2' as the union of all sets R_p with $p \in P_2$ and N' as the union of N'' and all newly introduced letters A_p, B_p' with $p \in P_2$.

By construction G' is a context-sensitive grammar since the rules of R_p replace only one letter and context is not changed. Thus it remains to prove that $L(G') = L(G'') = L(G)$

holds. The proof can be given analogously to the proof of Lemma 2.17. But since the proof is technical more involved we omit a complete formal proof. □

By Lemma 2.20, we obtain immediately that $\mathcal{L}(MON) \subseteq \mathcal{L}(CS)$ which gives with Lemma 2.14 the equality of the families of monotone and context-sensitive languages.

Theorem 2.21 *$\mathcal{L}(MON) = \mathcal{L}(CS)$.* □

If we start with a context-free grammars, then the above constructions to get the Kuroda normal form lead to a context-free grammar where all rules have the form

$$A \to B,\ A \to BC,\ A \to a,\ A \to \lambda, \tag{2.5}$$

where A, B, C are nonterminals and a is from the set of terminals. We now prove that also rules of the form $A \to B$ (which are called *chain rules*) and $A \to \lambda$ (which are called *erasing rules*) can be eliminated if we allow the exception which is used for monotone and context-sensitive grammars.

Lemma 2.22 *For any context-free grammar $G = (N, T, P, S)$ of order n, we can construct a context-free grammar $G' = (N', T, P', S')$ in time $O(2^n k(G) + k(G)^2)$ such that*
i) *P' contains no rule of the form $A \to \lambda$ with $A \neq S'$,*
ii) *$|w|_{S'} = 0$ for all rules $A \to w$ of P,*
iii) *$L(G') = L(G)$, and*
iv) *$k(G') \leq 2^n k(G)$.*

Proof. Let $G = (N, T, P, S)$ be a context free grammar. We first construct a context-free grammar $G'' = (N'', T, P'', S')$ which satisfies the conditions ii) and iii). For this purpose we add a new nonterminal S' to N, i. e., we set $N'' = N \cup \{S'\}$, and define the set of rules by $P'' = P \cup \{S' \to S\}$. Then ii) holds by construction and iii) is valid since all derivations in G'' have the form $S' \Longrightarrow S \overset{*}{\Longrightarrow} w$ where $S \overset{*}{\Longrightarrow} w$ is a derivation in G.

Let

$$M = \{A \mid A \in N'',\ A \overset{*}{\Longrightarrow} \lambda\}.$$

With any rule

$$q'' = A \longrightarrow v_1 A_1 v_2 A_2 \dots v_m A_m v_{m+1} \tag{2.6}$$

with

$$m \geq 0,\ A_1, A_2, \dots, A_m \in N'',\ v_1, v_2, \dots, v_{m+1} \in T^*$$

we associate the set $P_{q''}$ of all rules of the form

$$A \longrightarrow v_1 X_1 v_2 X_2 \dots v_m X_m v_{m+1} \neq \lambda$$

where

$$X_i = A_i \text{ for } A_i \notin M \quad \text{and} \quad X_i \in \{A_i, \lambda\} \text{ for } A_i \in M \quad \text{for} \quad 1 \leq i \leq m.$$

We note that $P_{q''} = \{q''\}$ if the right hand side of q'' is a non-empty terminal word. By definition, $P_{q''}$ does not contain a rule of the form $Y \to \lambda$. Therefore it is not possible to generate the empty word by rules of $P_{q''}$. Hence we set

$$\overline{P} = \begin{cases} \{S' \longrightarrow \lambda\} & \text{if } S' \in M \\ \emptyset & \text{otherwise} \end{cases}.$$

Moreover, we define $G' = (N', T, P', S')$ by

$$N' = N'' \quad \text{and} \quad P' = \overline{P} \cup \bigcup_{q'' \in P''} P_{p''}.$$

We mention that the property ii) is preserved and that P' satisfies condition i). We now prove that iii) is also valid. Clearly, it is sufficient to prove that $L(G') = L(G'')$.

First, by induction on the number of derivation steps, we prove that, for any nonterminal A and any word $x \in T^+$, $A \overset{*}{\underset{G''}{\Longrightarrow}} x$ implies $A \overset{*}{\underset{G'}{\Longrightarrow}} x$.

Let $n = 1$. In both grammars $A \Longrightarrow x$ is valid if we apply the rule $A \rightarrow x$, $x \in T^+$. This rule exist in P'' as well as in P'. Thus the induction basis is shown.

Let x be derived from A in n steps, $n \geq 2$. Then we have a derivation

$$A \underset{G''}{\Longrightarrow} v_1A_1v_2A_2 \ldots v_mA_mv_{m+1} \overset{*}{\underset{G''}{\Longrightarrow}} v_1x_1v_2x_2 \ldots v_mx_mv_{m+1} = x,$$

where, for $1 \leq i \leq m$, the derivation $A_i \overset{*}{\underset{G''}{\Longrightarrow}} x_i$ consist of less than n steps. We distinguish two cases for each i, $1 \leq i \leq n$.

Case 1. $x_i \neq \lambda$. Then we set $X_i = A_i$ and have a derivation $X_i \overset{*}{\underset{G'}{\Longrightarrow}} x_i$ by induction hypothesis.

Case 2. $x_i = \lambda$. Then we get $A_i \in M$ and we set $X_i = \lambda$.

By definition of P', we have the rule $A \rightarrow v_1X_1v_2X_2 \ldots v_mX_mv_{m+1}$ in P' and the derivation

$$A \underset{G'}{\Longrightarrow} v_1X_1v_2X_2 \ldots v_mX_mv_{m+1} \overset{*}{\underset{G'}{\Longrightarrow}} v_1x_1v_2x_2 \ldots v_mx_mv_{m+1}$$

in G', where we use $X_i = \lambda$ for $x_i = \lambda$ and the derivation $X_i \overset{*}{\underset{G'}{\Longrightarrow}} x_i$ for $x_i \neq \lambda$.

If we take the statement for $A = S$, we obtain that any word of $L(G'')$ different from the empty word can be generated by G'. Therefore we have $L(G'') \setminus \{\lambda\} \subseteq L(G') \setminus \{\lambda\}$. By the choice of $\overline{P}$, $\lambda \in L(G'')$ if and only if $\lambda \in L(G')$. Thus $L(G'') \subseteq L(G')$.

We now prove – by induction, again – that the converse implication also holds, i. e., for any terminating derivation $A \overset{*}{\underset{G'}{\Longrightarrow}} y$ in G', there is a terminating derivation $A \overset{*}{\underset{G''}{\Longrightarrow}} y$ in G''.

The induction basis can be shown as above.

Let $A \overset{*}{\underset{G'}{\Longrightarrow}} y$ be a derivation with n steps, $n \geq 2$. Then

$$A \Longrightarrow v_1X_1v_2X_2 \ldots v_mX_mv_{m+1} \overset{*}{\underset{G'}{\Longrightarrow}} v_1x_1v_2x_2 \ldots v_mx_mv_{m+1},$$

where $x_i = \lambda$ for $X_i = \lambda$ and $X_i \overset{*}{\underset{G'}{\Longrightarrow}} x_i$ is a derivation in G' with less than n derivation steps for $X_i \neq \lambda$. By the construction of $A \rightarrow v_1X_1v_2X_2 \ldots v_mX_mv_{m+1}$ in P', we have

– for $X_i = \lambda$, $A_i \in M$ and thus a derivation $A_i \overset{*}{\underset{G''}{\Longrightarrow}} \lambda = x_i$ in G'',

– for $X_i \neq \lambda$, a derivation $X_i = A_i \overset{*}{\underset{G''}{\Longrightarrow}} x_i$ in G'' by induction hypothesis.

Hence we have the derivation

$$A \underset{G''}{\Longrightarrow} v_1A_1v_2A_2 \ldots v_mA_mv_{m+1} \overset{*}{\underset{G'}{\Longrightarrow}} v_1x_2v_2x_2 \ldots v_mx_mv_{m+1}$$

in G''. Now we prove as above that $L(G') \subseteq L(G'')$.

By a combination of the two shown inclusions we get $L(G') = L(G'')$.

The statement iv) follows by the following facts. We add one nonterminal S' and at most two rules ($S' \to S$ and $S' \to \lambda$). Moreover, let q'' be a rule as given in (2.6) with s occurrences of nonterminals of M on the right hand side. Then $P_{q''}$ consists of 2^s rules, and each rule of $P_{q''}$ has smaller length than q''. Since s is bounded by the degree of the grammar, it is easy to see that $k(G') \le 2^n k(G)$ (the reader should verify this fact in the case that $M = \emptyset$, too).

Obviously, the construction of G' from G can be done in at most $2^n k(G)$ steps, too.

To complete the proof we have to show that there is an algorithm which determines M and works in a time $O(k(G)^2)$.

We put

$$M_0 = \emptyset \quad \text{and} \quad P_0 = P$$

and, for $i \ge 1$,

$$\begin{aligned} M_i &= M_{i-1} \cup \{A \mid A \in N'', A \to \lambda \in P_{i-1}\}, \\ P_i &= \{A \to w_1 w_2 \dots w_{n+1} \mid A \to w_1 A_1 w_2 A_2 \dots w_n A_n w_{n+1} \in P_{i-1} \\ &\qquad n \ge 0, w_j \in (N'' \setminus M_i)^* \text{ for } 1 \le j \le n+1, A_j \in M_i \text{ for } 1 \le j \le n\}. \end{aligned}$$

We first show by induction that $M_i \subseteq M$ for $i \ge 0$. For $i = 0$ and $i = 1$ the statement is obvious. For $A \in M_i$, $i \ge 2$, by definition of M_i, there is a rule $A \to A_1 A_2 \dots A_n$ with $A_j \in M_{i-1}$ for $1 \le j \le n$. Since $A_j \in M$ by induction hypothesis, we have a derivation

$$A \Longrightarrow A_1 A_2 \dots A_n \overset{*}{\Longrightarrow} \lambda A_2 A_3 \dots A_n \overset{*}{\Longrightarrow} \lambda\lambda A_3 \dots A_n \overset{*}{\Longrightarrow} \lambda^n = \lambda,$$

which implies $A \in M$.

Let $A \in M$. We consider a derivation $A \overset{*}{\Longrightarrow} \lambda$. None of the sentential forms of this derivation contains a terminal. Thus all sentential forms are words over N''. By a change of the order of the applications of the rules we can get a derivation

$$A = w_0 \overset{*}{\Longrightarrow} w_1 \overset{*}{\Longrightarrow} w_2 \overset{*}{\Longrightarrow} \dots \overset{*}{\Longrightarrow} w_m = \lambda$$

where, for $0 \le j \le m-1$, any subderivation $w_j \overset{*}{\Longrightarrow} w_{j+1}$ consist in a replacement of any nonterminal occurring in W_i according to a rule. Then all nonterminals which occur in w_{m-1} are replaced by λ, i. e., $w_{m-1} \in M_1^+$. Furthermore, any nonterminal A occurring in w_{m-2} is replaced by a non-empty word over M_1 or by λ. Hence $A \in M_2$ or $A \in M_1$. Since $M_1 \subseteq M_2$ by definition, $A \in M_2$, and consequently, $w_{m-2} \in M_2^+$. Continuing in this way, we get $w_{m-3} \in M_3^+$, $w_{m-4} \in M_4^+$ and finally $A = w_0 = w_{m-m} \in M_m$. Hence any element of M is contained in some M_i, $i \ge 1$.

Taking into consideration $M_i \le M$ for $i \ge 1$, we get

$$M = \bigcup_{i \ge 0} M_i.$$

According to the definitions, $M_i = M_{i+1}$ implies $P_i = P_{i+1}$ and then we obtain

$$M_i = M_{i+1} = M_{i+2} = \dots \quad \text{and} \quad P_i = P_{i+1} = P_{i+2} = \dots$$

Let $\#(N'') = t$. Then $\#(M) \leq t$. Because $M_i \subseteq M_{i+1}$ for $i \geq 0$, we obtain $M_t = M_{t+1}$ and therefore

$$M_t = \bigcup_{i \geq 0} M_i = M.$$

The construction M_0 and P_0 requires $\#(P) + 1$ steps. Moreover, to construct M_i requires to look on all rules of P_{i-1} and to add at most $\#(N)$ elements to M_{i-1}. The construction of P_i requires some cancellations of letters in rules of P and can therefore be performed in at most $k(G)$ steps. Because we stop with M_t, the construction of M requires at most the time $\#(P) + 1 + \#(N) \cdot (\#(P) + \#(N) + k(G)) \leq 2 \cdot k(G)^2$. $\square$

We illustrate the construction by an example, again. We consider the context-free grammar

$$G_9 = (\{S, A, B\}, \{a, b\}, \{S \to SA,\ S \to \lambda,\ A \to aAb,\ A \to B,\ B \to \lambda\}, S).$$

We note that

$$L(G_9) = \{a^{n_1}b^{n_1}a^{n_2}b^{n_2} \dots a^{n_k}b^{n_k} : k \geq 0,\ n_i \geq 0,\ 1 \leq i \leq k\}$$

since the first two rules generate an arbitrary number of A's and any A produces a word $a^n b^n$, $n \geq 0$. We get

$$\begin{aligned}
&N'' = N \cup \{S'\} = \{S, A, B, S'\},\\
&P'' = \{S' \to S, S \to SA, S \to \lambda, A \to aAb, A \to B, B \to \lambda\}\\
&M_0 = \emptyset \text{ and } P_0 = P'',\\
&M_1 = \{S, B\} \text{ and } P_1 = \{S' \to \lambda, S \to A, S \to \lambda, A \to aAb, A \to \lambda, B \to \lambda\},\\
&M_2 = \{S, B, S', A\} = N''\\
&N' = N'' = \{S', S, A, B\},\\
&\overline{P} = \{S' \to \lambda\},\\
&P' = \overline{P} \cup \{S' \to S, S \to SA, S \to A, S \to S, A \to aAb, A \to ab, A \to B\}\\
&\qquad \{S' \to \lambda, S' \to S, S \to SA, S \to A, S \to S, A \to aAb, A \to ab, A \to B\}.
\end{aligned}$$

We note that P' contains some superfluous rules. Since the application of $S \to S$ does not change the sentential form and there is no rule with left hand side B, we can omit the rules $S \to S$ and $A \to B$. Thus we get the grammar

$$G_9' = (\{S', S, A\}, \{a, b\}, \{S' \to \lambda, S' \to S, S \to SA, S \to A, A \to aAb, A \to ab\}, S').$$

It is obvious, that all rules of the grammar G' constructed in (the proof of) Lemma 2.22 – perhaps with exception of $S' \to \lambda$ – are of the form $A \to w$ with a non-empty word w. Thus these rules are monotone. Moreover, by construction (see condition ii), S' does not occur on the right hand side of rules of P'. Consequently, any G' is a monotone grammar. Thus we can reformulate Lemma 2.22 as follows: *For any context-free grammar G, there is a monotone grammar G' such that $L(G') = L(G)$.* By Lemma 2.13, we obtain the following statement.

Theorem 2.23 *$\mathcal{L}(CF) \subseteq \mathcal{L}(MON)$.* $\square$

If we combine Lemma 2.14, Theorem 2.21 and Theorem 2.23 we get the hierarchy

$$\mathcal{L}(REG) \subseteq \mathcal{L}(LIN) \subseteq \mathcal{L}(CF) \subseteq \mathcal{L}(MON) = \mathcal{L}(CS) \subseteq \mathcal{L}(RE) \tag{2.7}$$

Thus it remains to discuss the properness of the inclusions which will be done in the following section concerning the first three inclusions and in Chapter **??** for the last one. For the corresponding proofs we need further normal form results.

Lemma 2.24 *For any context-free grammar $G = (N, T, P, S)$ of order n, a context-free grammar $G' = (N, T, P', S)$ can be constructed in time $O(n \cdot k(G)^2)$ such that*
– *P' contains no rules of the form $A \to B$ with $A \in N$ and $B \in N$ and*
– *$L(G') = L(G)$, and*
– *$k(G') \in O(n \cdot k(G)^2)$.*

Proof. For a nonterminal $A \in N$, we set

$$M_A = \{B \mid B \Longrightarrow C_1 \Longrightarrow C_2 \Longrightarrow \dots C_k \Longrightarrow A,\ A, B, C_1, C_2, \dots, C_k \in N, k \geq 0\} \cup \{A\}$$

(note that $A \in M_A$ by definition). For any rule $p = A \to w$ with $w \notin N$, we set

$$P_p = \{B \to w \mid B \in M_A\},$$

i. e., we replace a derivation

$$B \Longrightarrow C_1 \Longrightarrow C_2 \Longrightarrow \dots \Longrightarrow C_k = A \Longrightarrow w$$

by the rule $B \to w$. We define

$$P' = \bigcup_{p \in P} P_p.$$

Obviously, $G' = (N, T, P', S)$ satisfies all conditions required in the assertion. The validity of $L(G') = L(G)$ can be shown analogously to the proof of Lemma 2.22.

Obviously, for any rule $p = A \to w$, the added rules of P_p contribute $(|w| + 3)\#(M_A)$ to the length of the description of G'. Since we have to consider at most $\#(P)$ rules, $\#(M_A) \leq \#(N)$ for all $A \in N$ and $|w| \leq n$ for all rules $A \to w \in P$, we get $k(G') \in O(n \cdot k(G)^2)$.

Thus G' can be written in time $O(n \cdot k(G)^2)$ and to determine the time of the construction we have to add the time needed to construct the sets M_A for all $A \in N$.

We construct the graph $H = (N, E)$, where the nonterminals are the nodes and there is an edge from B to C if and only if there is a rule $C \to B$ in P. Obviously, H has at most $\#(P)$ edges. Then it is easy to determine M_A for a given A by depth first search or breadth first search in time $O(\#(N) + \#(P))$ and hence in $O(k(G))$.

Since we have to determine the sets M_A for all $A \in N$, the construction of the sets M_A can be done in time $O(k(G)^2)$. □

We apply the construction given in the proof of Lemma 2.24 to G_9' constructed above. We obtain

$$M_{S'} = \{S'\},\ M_S = \{S, S'\} \text{ and } M_A = \{A, S, S'\}$$

and thus

$$\begin{aligned}
P_{S'\to\lambda} &= \{S'\to\lambda\}\,,\\
P_{S\to SA} &= \{S\to SA, S'\to SA\}\,,\\
P_{A\to aAb} &= \{A\to aAb, S\to aAb, S'\to aAb\}\,,\\
P_{A\to ab} &= \{A\to ab, S\to ab, S'\to ab\}
\end{aligned}$$

and finally $G_9'' = (\{S', S, A\}, \{a, b\}, P_9'', S')$ with

$$\begin{aligned}
P_9'' = \{&S'\to\lambda, S\to SA, S'\to SA, A\to aAb, S\to aAb, S'\to aAb,\\
&A\to ab, S\to ab, S'\to ab\}.
\end{aligned}$$

If we combine all the constructions of normal forms of context-free grammars, then we obtain a context-free grammar which contains only rules of the form $A\to BC$ or $A\to a$ with nonterminals A, B, C and a from the set of terminals since we have removed the chain rules and erasing rules from the allowed rules in (2.5).

Definition 2.25 *A context-free grammar $G = (N, T, P, S)$ is in Chomsky normal form if all rules of P have the form $A\to BC$ or $A\to a$, where A, B, C are from the set of nonterminals and a is from the set of terminals, with the possible exception of $S\to\lambda$ if S does not occur on the right side of any production of P*

The remark before Definition 2.25 can be be reformulated as follows.

Theorem 2.26 *For any context-free grammar G, a context-free grammar G' in Chomsky normal form can be constructed in a time $O(k(G)^2)$ such that $L(G') = L(G)$ and $k(G')\in O(k(G)^2)$.*

Proof. By applying the constructions of the Lemmas 2.15 – 2.17, we construct a context-free grammar G'' of order 2, i.e., all rules have the forms given in (2.5). Since we start with a context-free grammar, we have to apply only (2.2). Thus a rule $A\to B_1B_2\dots B_n$ is transformed into the rules

$$A\to B_1C_1,\ C_1\to B_2C_2,\dots,C_{n-2}\to B_{n-2}C_{n-1},\ C_{n-1}\to B_{n-1}B_n$$

. Thus we get $k(G'')\in O(k(G))$ and the transformation can be done in a time $O(k(G))$.

Now we apply Lemma 2.22 to G'' and eliminated all erasing rules. Because G'' has order 2, the obtained grammar G''' can be constructed in time $O(k(G'')^2) = O(k(G)^2)$ and has size in $O(k(G'')) = O(k(G))$.

Finally we construct G' by eliminating the chain rules according to Lemma 2.24. Again, since G''' has order 2, G' is constructed in time $O(k(G)^2)$ and satisfies $k(G')\in O(k(G)^2)$. □

In order to get G_9'' in Chomsky normal form we have to transform the rules containing terminals in their right hand sides and to shorten the lengths of the right hand side of some rules. This can be done by the constructions given in the Lemmas 2.15 and 2.17, which gives in succession the following sets of productions

$$\begin{aligned}
&\{S' \to \lambda, S \to SA, S' \to SA, A \to a'Ab', S \to a'Ab', S' \to a'Ab',\\
&\quad A \to a'b', S \to a'b', S' \to a'b', a' \to a, b' \to b\}\\
&\{S' \to \lambda, S \to SA, S' \to SA, A \to a'A_1, A_1 \to Ab', S \to a'A_2, A_2 \to Ab',\\
&\quad S' \to a'A_3, A_3 \to Ab', A \to a'b', S \to a'b', S' \to a'b', a' \to a, b' \to b\}.
\end{aligned}$$

The reader may verify that it is not necessary to use three different letters A_1, A_2 and A_3 (they are different by construction since they belong to different rules).

The normal forms given above do not exist for linear and regular grammars since they need at least a rule $A \to BC$ with nonterminals A, B, C if the generated contains at least one word of length ≥ 2 and such rules are not allowed in linear and regular grammars. Thus we now present normal forms for linear and regular grammars.

Theorem 2.27 *For any linear grammar $G = (N, T, P, S)$, a linear grammar $G' = (N', T, P', S')$ can be constructed in time $O(k(G)^2)$ such that*

- *P' contains only rules of the forms $A \to aB$ or $A \to Ba$ or $A \to a$ with $A, B \in N'$ and $a \in T$ with the possible exception $S' \to \lambda$ if S' does not occur on the right side of any production of P', and*
- *$L(G') = L(G)$, and*
- *$k(G') \in O(k(G)^2)$.*

Proof. Let $G = (N, T, P, S)$ be a linear grammar. Let P_1 be the set of all rules of P of the form $A \to uBv$ with $|uv| \geq 2$ and $A \to w$ with $|w| \geq 2$. Note that all rules of P not contained in P_1 have the form

$$A \to aB \text{ or } A \to Ba \text{ or } A \to B \text{ or } A \to a \text{ or } A \to \lambda \tag{2.8}$$

with $A, B \in N$ and $a \in T$.

For any rule $p = A \to a_1 a_2 \dots a_n B b_1 b_2 \dots b_m \in P_1$ with $n \geq 1$ and $m \geq 1$, $a_i \in T$ for $1 \leq i \leq n$ and $b_j \in T$ for $1 \leq j \leq m$, we introduce new nonterminals

$$A_{p,1}, A_{p,2}, \dots, A_{p,n-1}, B_{p,1}, B_{p,2}, \dots, B_{p,m}$$

and new rules

$$\begin{aligned}
&A \to a_1 A_{p,1},\ A_{p,1} \to a_2 A_{p,2},\ A_{p,2} \to A_{p,3}, \dots,\ A_{p,n-2} \to a_{n-1} A_{p,n-1},\ A_{p,n-1} \to a_n B_{p,m},\\
&B_{p,m} \to b_{p,m-1} b_m,\ B_{p,m-1} \to B_{p,m-2} b_{m-1}, \dots,\ B_{p,2} \to B_{p,1} b_2,\ B_{p,1} \to B a_1.
\end{aligned}$$

Let N_p and P_p be the sets of new nonterminals and new rules, respectively, associated with p. If $n = 0$ and $m \geq 2$ or $n \geq 2$ and $m = 0$, we modify the construction of N_p and P_p by taking $B_{p,m} = A$ or $B_{p,m} = B$.

For any rule $q = A \to a_1 a_2 \dots a_n \in P_1$ with $n \geq 2$ and $a_i \in T$ for $1 \leq i \leq n$, let

$$\begin{aligned}
N_q &= \{A_{q,1},\ A_{q,2}, \dots,\ A_{q,n-1}\},\\
P_q &= \{A \to a_1 A_{q,1},\ A_{q,1} \to a_2 A_{q,2}, \dots,\ A_{q,n-1} \to a_{n-1} A_{q,n-1},\ A_{q,n-1} \to a_n\}.
\end{aligned}$$

We now construct the grammar $G'' = (N'', T, P'', S)$ with

$$N'' = N \cup \bigcup_{p \in P_1} N_p, \quad P'' = (P \setminus P_1) \cup \bigcup_{p \in P_1} P_p.$$

Clearly, all productions in P'' have the form given in (2.8).

By the construction it is easy to see that $k(G'') \in O(k(G))$ and the construction can be done in time $O(k(G))$.

Let $p = A \to a_1a_2 \dots a_nBb_1b_2 \dots b_m \in P_1$, $n \geq 0$, $m \geq 0$, $n+m \geq 2$, $a_i \in T$ for $1 \leq i \leq n$ and $b_j \in T$ for $1 \leq j \leq m$, and let

$$xAy \Longrightarrow xa_1a_2 \dots a_nBb_1b_2 \dots b_my \tag{2.9}$$

be a derivation step in G obtained by an application of p. Then

$$\begin{aligned} xAy &\Longrightarrow xa_1A_{p,1}y \Longrightarrow xa_1a_2A_{p,2}y \Longrightarrow \dots \Longrightarrow xa_1 \dots a_{n-1}A_{p,n}y \\ &\Longrightarrow xa_1 \dots a_nB_{p,m}y \Longrightarrow xa_1 \dots a_nB_{p,m-1}b_1y \Longrightarrow \dots \Longrightarrow xa_1 \dots a_nB_{p,1}b_2 \dots b_my \\ &\Longrightarrow xa_1 \dots a_nBb_1b_2 \dots b_my \end{aligned} \tag{2.10}$$

is a derivation in G'' by applications of the rules of P_p in the order given above.

Analogously, if $q = A \to a_1a_2 \dots a_n \in P_2$, $a_i \in T$ for $1 \leq i \leq n$, and

$$xAy \Longrightarrow xa_1a_2 \dots a_ny \tag{2.11}$$

is a derivation step in G according to q, then

$$xAy \Longrightarrow xa_1A_{q,1}y \Longrightarrow xa_1a_2A_{q,2}y \Longrightarrow \dots \Longrightarrow xa_1a_2 \dots a_{n-1}A_{q,n-1}y \Longrightarrow xa_1a_2 \dots a_ny \tag{2.12}$$

is a derivation in G''.

Let

$$\begin{aligned} S = A_0 &\Longrightarrow u_1A_1v_1 \Longrightarrow u_1u_2A_2v_2v_1 \Longrightarrow \dots \\ &\Longrightarrow u_1u_2 \dots u_rA_rv_rv_{r-1} \dots v_1 \Longrightarrow u_1u_2 \dots u_rwv_rv_{r-1} \dots v_1 \end{aligned} \tag{2.13}$$

be a derivation of a terminal word in G. If we replace any derivation step in (2.13) which is done by an application of a rule of P_1, i.e., it is of type (2.9) or (2.11), by the corresponding derivation (2.10) or (2.12), respectively, and do not change derivation steps according to rules of $P \setminus P_1$, then we obtain a derivation

$$\begin{aligned} S = A_0 &\overset{*}{\Longrightarrow} u_1A_1v_1 \overset{*}{\Longrightarrow} u_1u_2A_2v_2v_1 \overset{*}{\Longrightarrow} \dots \\ &\overset{*}{\Longrightarrow} u_1u_2 \dots u_rA_rv_rv_{r-1} \dots v_1 \overset{*}{\Longrightarrow} u_1u_2 \dots u_rwv_rv_{r-1} \dots v_1 \end{aligned} \tag{2.14}$$

in G''. Thus $L(G) \subseteq L(G'')$.

Conversely, if we have a sentential form xAv of G' and apply a rule of P_p for some $p \in P_1$, then it is the rule $A \to a_1A_{p,1}$ if $n \geq 1$ or $A \to B_{p,m-1}b_m$ if $n = 0$ and have to apply the rules in the order given above, i.e., we get a derivation of the form (2.10) or (2.12). Thus, any derivation in G'' is of the form (2.14) where $A_{i-1} \to u_iA_iv_i$ is a rule in G.

Furthermore, if we replace in (2.14) any subderivation $x_{i-1}A_{i-1}y_{i-1} \overset{*}{\Longrightarrow} x_{i-1}u_iA_{i-1}v_iy_{i-1}$ of the form (2.10) and $x_rA_ry_r \overset{*}{\Longrightarrow} x_rwy_{i-1}$ of the form (2.12) by its corresponding derivation step (2.9) or (2.11), respectively, we obtain a derivation (2.13) in G. Hence $L(G'') \subseteq L(G)$, too.

Consequently, $L(G'') = L(G)$.

Now we eliminate all rules of the forms $A \to \lambda$ and $A \to B$ (such rules are in $P \setminus P_1$) by the constructions given in the proofs of Lemmas 2.22 and 2.24. It is easy to see that these constructions preserve the linearity of the grammar. Let $G' = (N', T, P', S')$ be the linear grammar which we obtain by these constructions. Then all rules of P' are of the form $A \to aB$ or $A \to Ba$ or $A \to a$ with $A, B \in N'$ and $a \in T$ with the possible exception $S' \to \lambda$ if S' does not occur on the right side of any production of P'. Moreover, as in the proofs of Lemmas 2.22 and 2.24, we can show that $L(G') = L(G'') = L(G)$.

As in the proof of Theorem 2.26 we get $k(G') \in O(k(G)^2)$ and G' can be constructed in time $O(k(G)^2)$. □

Theorem 2.28 *For any regular grammar $G = (N, T, P, S)$, a regular grammar $G' = (N', T, P', S')$ can be constructed in time $O(k(G^2))$ such that*

- *P' contains only rules of the forms $A \to aB$ or $A \to a$ with $A, B \in N'$ and $a \in T$ with the possible exception $S \to \lambda$ if S does not occur on the right side of any production of P', and*
- *$L(G') = L(G)$, and*
- *$k(G') \in O(k(G)^2)$*

Proof. The proof can be given by applying the construction given in the proof of Lemma 2.27 for the special case of regular rules. □

By the construction the regular grammar

$$G_{10} = (\{S\}, \{a, b, c\}, \{p, q, r\}, S) \text{ with } p = S \to abS,\ q = S \to acS,\ r = S \to abc$$

generating the language

$$L(G_{10}) = \{ax_1ax_2 \dots ax_nabc \mid n \geq 0, x_i \in \{a, b\} \text{ for } 1 \leq i \leq n\}$$

is transformed in the normal form grammar

$$G'_{10} = \{S, A_{p,1}, A_{q,1}, A_{r,1}, A_{r,2}\}, \{a, b, c\}, P, S$$

with

$$P = \{S \to aA_{p,1}, A_{p,1} \to bS, S \to aA_{q,1}, A_{q,1} \to cS, S \to aA_{r,1}, A_{r,1} \to bA_{r,2}, A_{r,2} \to c\}.$$

In all the normal forms we have given hitherto we have made a restriction of the allowed rules. In context-free grammars one can also consider a restriction concerning the position where the rule is applied.

Definition 2.29 *Let $G = (N, T, P, S)$ be a context-free grammar. A derivation step $\gamma \Longrightarrow \gamma'$ is called* leftmost *if*

– *$\gamma = \alpha A \beta$ with $\alpha \in T^*$, $A \in N$ and $\beta \in (N \cup T)^*$,*
– *$\gamma' = \alpha w \beta$ for some rule $A \to w \in P$.*

In case of a leftmost derivation we write $\gamma \underset{l}{\Longrightarrow} \gamma'$.

The language accepted by G by leftmost derivations is

$$L_l(G) = \{w \mid w \in T^*,\ S \overset{*}{\underset{l}{\Longrightarrow}} w\},$$

where $\overset{*}{\underset{l}{\Longrightarrow}}$ is the reflexive and transitive closure of $\underset{l}{\Longrightarrow}$.

By Definition 2.29, in leftmost derivation we replace the leftmost occurence of a nonterminal in the current sentential form. The following result states that leftmost derivations can be considered as a normal form with respect to the position where we apply the rules.

Theorem 2.30 *For any context-free grammar G, $L(G) = L_l(G)$.*

Proof. We prove the following statement by induction on the length of the derivations: For any $w \in T^*$,

$$z \overset{*}{\Longrightarrow} w \text{ if and only if } z \overset{*}{\underset{l}{\Longrightarrow}} w \tag{2.15}$$

If the number of derivation steps is one, then z contains exactly one nonterminal A and a rule $A \to v$ with $v \in T^*$ is applied. Therefore the only possible derivation is a leftmost one.

Now let $z = \alpha A \beta$ with $\alpha \in T^*$, and let $z \overset{*}{\Longrightarrow} w$ be done by n direct derivation steps. We have to apply in a certain derivation step a rule $A \to v$ to A. Let $\beta \overset{*}{\Longrightarrow} \beta'$ be the derivation which is performed before the use of $A \to v$. Then we have the derivation

$$\alpha A \beta \overset{*}{\Longrightarrow} \alpha A \beta' \Longrightarrow \alpha v \beta' \overset{*}{\Longrightarrow} w.$$

Then there also is a derivation

$$\alpha A \beta \overset{*}{\Longrightarrow} \alpha v \beta \Longrightarrow \alpha v \beta' \overset{*}{\Longrightarrow} w.$$

Obviously, the derivation $v\beta \overset{*}{\Longrightarrow} v\beta' \overset{*}{\Longrightarrow} w$ can be done in $n-1$ direct derivation steps. By induction assumption we know that there is a leftmost derivation $v\beta' \overset{*}{\underset{l}{\Longrightarrow}} w$. Hence we have the leftmost derivation $z = \alpha A \beta \underset{l}{\Longrightarrow} \alpha v \beta \overset{*}{\underset{l}{\Longrightarrow}} w$.

Thus, for any derivation $z \overset{*}{\Longrightarrow} w$, there is a leftmost derivation $z \overset{*}{\underset{l}{\Longrightarrow}} w$. The converse direction also holds, because any leftmost derivation is a (usual) derivation. Therefore (2.15) holds.

As a special case of (2.15) we get $S \overset{*}{\Longrightarrow} w$ if and only if $S \overset{*}{\underset{l}{\Longrightarrow}} w$ which proves $L(G) = L_l(G)$. □

Let $w_1 A w_2 B w_3$ be a sentential form of a context-free G with rules $A \to x$ and $B \to y$. Then the existence of a derivation

$$w_1 A w_2 B w_3 \Longrightarrow w_1 x w_2 B w_3 \overset{*}{\Longrightarrow} w_1' x w_2' B w_3' \Longrightarrow w_1' x w_2' y w_3',$$

where $w_1 \overset{*}{\Longrightarrow} w_1'$, $w_2 \overset{*}{\Longrightarrow} w_2'$, and $w_3 \overset{*}{\Longrightarrow} w_3'$ are certain derivations according to the grammar, implies the existence of the derivation

$$w_1 A w_2 B w_3 \Longrightarrow w_1 A w_2 y w_3 \overset{*}{\Longrightarrow} w_1' A w_2' y w_3' \Longrightarrow w_1' x w_2' y w_3'.$$

This means that we change the order of the application of rules without a change of the generated language. Especially, we also generate $L(G)$, if we apply in any derivation step a rule to the leftmost nonterminal symbol in the current sentential form. This proves Theorem 2.30, again, but in a little bit more informal way.

2.3 Iteration Theorems

The aim of this section is to prove that the inclusions $\mathcal{L}(REG) \subseteq \mathcal{L}(LIN) \subseteq \mathcal{L}(CF) \subseteq \mathcal{L}(MON)$ are proper. For this purpose, for example with respect to the first inclusion we have to give a language L which is linear, but not regular. In order to show the linearity it is sufficient to give a linear grammar generating L. The problematic part is to verify that L is not regular, i. e., by definition, we have to prove that no regular grammar generates L. This is a hard task since there are infinitely many regular grammars. Thus we give a property which ensures that a regular language, which contains a certain word z, also contains some further words related to z. Now it is sufficient to give a linear language which does not have this property.

Theorem 2.31 *Let L be a regular language. Then there is a constant k (depending on L) such that, for any word z with $|z| \geq k$ there are words u, v, w with*
i) *$z = uvw$,*
ii) *$|uv| \leq k$, $|v| > 0$, and*
iii) *$uv^iw \in L$ for all $i \geq 0$.*

Proof. By Theorem 2.28, there is a regular grammar $G = (N, T, P, S)$ such that $L = L(G)$ and P contains only productions of the form $A \longrightarrow aB$ and $A \longrightarrow a$ with $A, B \in N$ and $a \in T$. We set $k = \#(N) + 1$. The possible exceptional rule $S \longrightarrow \lambda$ is only used to generate the empty word; and therefore we can ignore this rule since we only interested in words of length at most k, and we have chosen $k \geq 1$.

By the form of the rules of P, for any word $z = a_1a_2 \dots a_n$ with $a_i \in T$ for $1 \leq i \leq n$ and $n \geq k$, there is a derivation

$$\begin{aligned} S = A_0 &\Longrightarrow a_1A_1 \Longrightarrow a_1a_2A_2 \Longrightarrow a_1a_2a_3A_3 \Longrightarrow \dots \\ &\Longrightarrow a_1a_2 \dots a_{n-1}A_{n-1} \Longrightarrow a_1a_2 \dots a_{n-1}a_n = z. \end{aligned} \tag{2.16}$$

By the choice of k, the set $\{A_0, A_1, A_2, \dots, A_k\}$ contains at least one nonterminal twice. Let $A = A_i = A_j$ with that $0 \leq i < j \leq k$. We set

$$u = a_1a_2 \dots a_i,\ v = a_{i+1}a_{i+2} \dots a_j \text{ and } w = a_{j+1}a_{j+2} \dots a_n.$$

It is obvious that the conditions i) and ii) are satisfied.

The derivation (2.16) can be written in the form

$$S = A_0 \overset{*}{\Longrightarrow} uA \overset{*}{\Longrightarrow} uvA \overset{*}{\Longrightarrow} uvw = z,$$

and, for $i \geq 2$ and $i = 0$, we also have the derivations

$$S = A_0 \overset{*}{\Longrightarrow} uA \overset{*}{\Longrightarrow} uvA \overset{*}{\Longrightarrow} uvvA \overset{*}{\Longrightarrow} uvvvA \overset{*}{\Longrightarrow} \dots \overset{*}{\Longrightarrow} uv^iA \overset{*}{\Longrightarrow} uv^iw \in T^*$$

and

$$S \overset{*}{\Longrightarrow} uA \overset{*}{\Longrightarrow} uw \in T^*,$$

respectively. Therefore $uv^iw \in L(G) = L$ für $i \geq 0$, which proves our assertion. □

Theorem 2.31 is called *iteration theorem for regular languages* since its proof is based on the iteration of some subderivation. Sometimes it is also called *pumping lemma/theorem for regular languages* because the subword v of z is "pumped".

We now apply Theorem 2.31 to show the existence of a language in $\mathcal{L}(LIN) \setminus \mathcal{L}(REG)$. We consider the language

$$L = L(G_2) = \{a^nb^n \mid n \geq 1\},$$

where G_2 is the linear grammar given in Example 2.5. Thus $L \in \mathcal{L}(LIN)$.

Let us assume that L is regular. By Theorem 2.31, there is a constant k such that the properties given in Theorem 2.31 hold. Let $z = a^kb^k$. Since $|z| > k$, there exists a partition $z = uvw$ such that $|uv| \leq k$, $v \neq \lambda$ and $uv^iw \in L$ for all $i \geq 0$. By the first two mentioned properties, $v = a^r$ for some $r \geq 1$ and $u = a^s$ for some $s \geq 0$ and $w = a^{k-r-s}b^k$. Now we have $uv^2w \in L$ by the iteration theorem. But by the structure of the words in L (the number of occurrences of a equals the number of occurrences of b), $uv^2w = a^sa^{2r}a^{k-r-s}b^k = a^{k+r}b^k \notin L$. The obtained contradiction justifies that our assumption does not hold, i. e., L is not regular.

From the existence of L and Lemma 2.14, we get immediately the following fact.

Theorem 2.32 *The family $\mathcal{L}(REG)$ is properly included in $\mathcal{L}(LIN)$.* □

We give a iteration theorem or pumping lemma for linear languages and use it to prove $\mathcal{L}(LIN) \subset \mathcal{L}(CF)$.

Theorem 2.33 *Let L be a linear language. Then there exists a constant k (depending on L) such that, for any word $z \in L$ with $|z| \geq k$, there exist words u, v, w, x, y with*
i) *$z = uvwxy$,*
ii) *$|uvxy| \leq k$, $|vx| > 0$, and*
iii) *$uv^iwx^iy \in L$ for all $i \geq 0$.*

Proof. Let L be a linear language. By Theorem 2.27, there is a linear grammar $G = (N, T, P, S)$ such that $L = L(G)$ and each rule has the form $A \to uBv$ or $A \to w$ with $|uv| = |w| = 1$ (again, the exception $S \to \lambda$ is not of interest for us). We set

$$k = \#(N) + 1.$$

Let $z \in L$ and $|z| \geq k$. Let n be the number of direct derivation steps to obtain z from S. Then $n \geq k$ since each step adds at most one terminal letters. Thus there is a derivation

$$\begin{aligned} S = A_0 &\Longrightarrow u_1A_1v_1 \Longrightarrow u_1u_2A_2v_2v_1 \Longrightarrow \ldots \Longrightarrow u_1u_2\ldots u_{k_1}A_{k_1}v_{k_1}v_{k_1-1}\ldots v_1 \\ &\overset{*}{\Longrightarrow} u_1u_2\ldots u_{k_1}u_{k_1+1}\ldots u_{n-1}A_{n-1}v_{n-1}v_{n-2}\ldots v_{k_1}\ldots v_1 \\ &\Longrightarrow u_1u_2\ldots u_{k_1}u_{k_1+1}\ldots u_{n-1}z'v_{n-1}v_{n-2}\ldots v_{k_1}\ldots v_1 = z. \end{aligned}$$

Because N contains $k-1$ nonterminals, there are integers i and j with $0 \leq i < j \leq k_1$ and $A_i = A_j$. Then we set

$$\begin{aligned} u &= u_1 u_2 \dots u_i,\ y = v_i v_{i-1} \dots v_1, \\ v &= u_{i+1} u_{i+2} \dots u_j,\ x = v_j v_{j-1} \dots v_{i+1}, \\ w &= u_{j+1} u_{j+2} \dots u_{n-1} z' v_{n-1} v_{n-2} \dots v_{j+1}. \end{aligned}$$

Then we have

$$|uvxy| = |u_1 u_2 \dots u_j v_j v_{j-1} \dots v_1| \leq k \text{ and } z = uvwxy.$$

Therefore i) and ii) are satisfied. Moreover, we have the derivations

$$\begin{aligned} S &\overset{*}{\Longrightarrow} uA_iy \overset{*}{\Longrightarrow} uvA_jxy = uvA_ixy \overset{*}{\Longrightarrow} uvvA_jxxy = uvvA_ixxy \\ &\overset{*}{\Longrightarrow} uv^iA_jx^iy = uv^iwx^iy, \end{aligned}$$

which proves iii). □

Theorem 2.34 *The family $\mathcal{L}(LIN)$ is properly included in $\mathcal{L}(CF)$.*

Proof. Since the inclusion is known by (2.7), it is sufficient to show that the inclusion is strict.

We consider the context-free grammar

$$G = (\{S, A\},\ \{a, b\},\ \{S \to AA,\ A \to aAb,\ A \to ab\},\ S).$$

It is easy to see that

$$L(G) = \{a^n b^n a^m b^m \mid n \geq 1,\ m \geq 1\}. \tag{2.17}$$

Clearly, $L(G) \in \mathcal{L}(CF)$. We show that $L(G) \notin \mathcal{L}(LIN)$.

Let us assume that $L(G)$ is a linear language. Let k be the constant which exist by Theorem 2.33. We consider the word $z = a^{2k}b^{2k}a^{2k}b^{2k} \in L(G)$ of length $8k > k$. By Theorem 2.33, there exist a partition $z = uvwxy$ such that $|uvxy| \leq k$ and $uv^iwx^iy \in L(G)$ for $i \geq 0$. By $|uvxy| \leq k$, the words u and v only contain the letter a and y and x only contain the letter b, i.e., we have $u = a^r$, $v = a^t$, $x = b^p$ and $y = b^q$ for certain integers r, t, p, q and $z = a^r a^t a^{2k-r-t} b^{2k} a^{2k} b^{2k-p-q} b^p b^q$. Thus, for $i = 2$,

$$z' = a^r a^{2t} a^{2k-r-t} b^{2k} a^{2k} b^{2k-p-q} b^{2p} b^q = a^{2k+t} b^{2k} a^{2k} b^{2k+p} \in L(G). \tag{2.18}$$

By Theorem 2.33, in addition, $t > 0$ or $p > 0$ because $vx \neq \lambda$. This implies $z' = a^{2k+t}b^{2k}a^{2k}b^{2k+p} \notin L(G)$ by the structure of the words in $L(G)$ (see (2.17)) which contradicts (2.18). □

We now present the iteration theorem/pumping lemma for context-free languages.

Theorem 2.35 *Let L be a context-free language. Then there is a constant k (depending on L) such that, for any word $z \in L$ with $|z| \geq k$ there are words u, v, w, x, y with*

i) $z = uvwxy$,

ii) $|vwx| \leq k$, $|v| + |x| > 0$, *and*

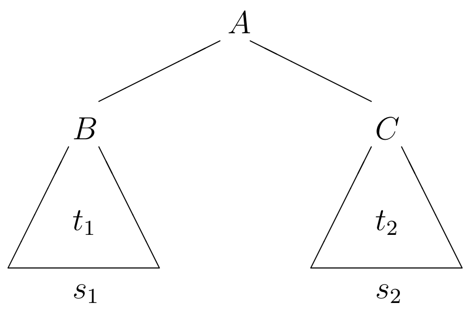
iii) $uv^iwx^iy \in L$ *for all* $i \geq 0$.

Proof. Let L be a context-free language. By Theorem 2.26, $L = L(G)$ for some context-free grammar $G = (N, T, P, S)$ in Chomsky normal form. Let $n = \#(N)$. We set $k = 2^n$.

Let $A \Longrightarrow^* s \in T^*$ be a derivation with an associated derivation tree of depth m. We first prove by induction on the depth m that $|s| < 2^m$ gilt.

$m = 1$. Then the derivation consists of a single step. By the definition of the Chomsky normal form, the derivation step is an application of a rule of the form $A \to a$ for some $a \in T$ or $A \to \lambda$ (if A is the axiom). Thus $s = \lambda$ oder $s = a$ which immediately gives $|s| \leq 1 < 2 = 2^1$. Thus the induction base is shown.

Let $A \overset{*}{\Longrightarrow} w$ be a derivation with a derivation tree t of depth $m \geq 2$. By the Chomsky normal form, t has the form



where t_1 and t_2 are derivation trees with a depth at most $m-1$ and $s_1s_2 = s$ holds. By induction hypothesis

$$|s| = |s_1| + |s_2| < 2^{m-1} + 2^{m-1} = 2^m,$$

which finishes the proof of the assertion.

We use this statement on derivation trees for words $z \in L$ with $|z| \geq k$. We obtain that the derivation tree t' associated with a derivation of z has a depth $m \geq n+1$ according to the choice of k. Therefore t' has the form shown in Figure 2.2.

Because $m - 1 \geq n$, there is a nonterminal which occurs twice in the sequence $S, A_1, A_2, \ldots, A_{m-1}$. Let A be this nonterminal. Then t' has the form shown in Figure 2.3.

Since G is in Chomsky normal form, $vx \neq \lambda$. Furthermore, without loss of generality we can assume that $|vwx| \leq k$, because otherwise there is a path in the derivation tree associated with $A \overset{*}{\Longrightarrow} vwx$ which contains a certain nonterminal A' at least two times and we can choose A' instead A. Thus the conditions i) and ii) are shown.

From t we can see that we have the derivations

$$S \overset{*}{\Longrightarrow} uAy, \qquad A \overset{*}{\Longrightarrow} vAx, \qquad A \overset{*}{\Longrightarrow} w.$$

Hence we also have the derivation

$$S \overset{*}{\Longrightarrow} uAy \overset{*}{\Longrightarrow} uvAxy \overset{*}{\Longrightarrow} uvvAxxy \overset{*}{\Longrightarrow} uvvvAxxxy \overset{*}{\Longrightarrow} \ldots \overset{*}{\Longrightarrow} uv^iAx^iy \overset{*}{\Longrightarrow} uv^iwx^iy$$

for $i \geq 0$ (if $i = 1$, then the given word z is derived). Because this derivation yield a word from T^*, we obtain $uv^iwx^iy \in L(G) = L$ for $i \geq 0$, which proves condition iii). $\square$

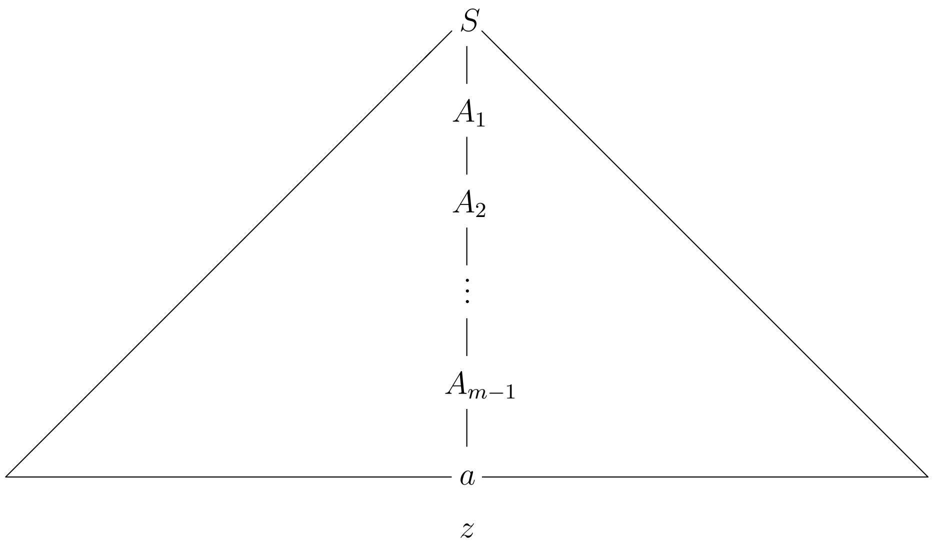


Figure 2.2: Derivation tree of height m

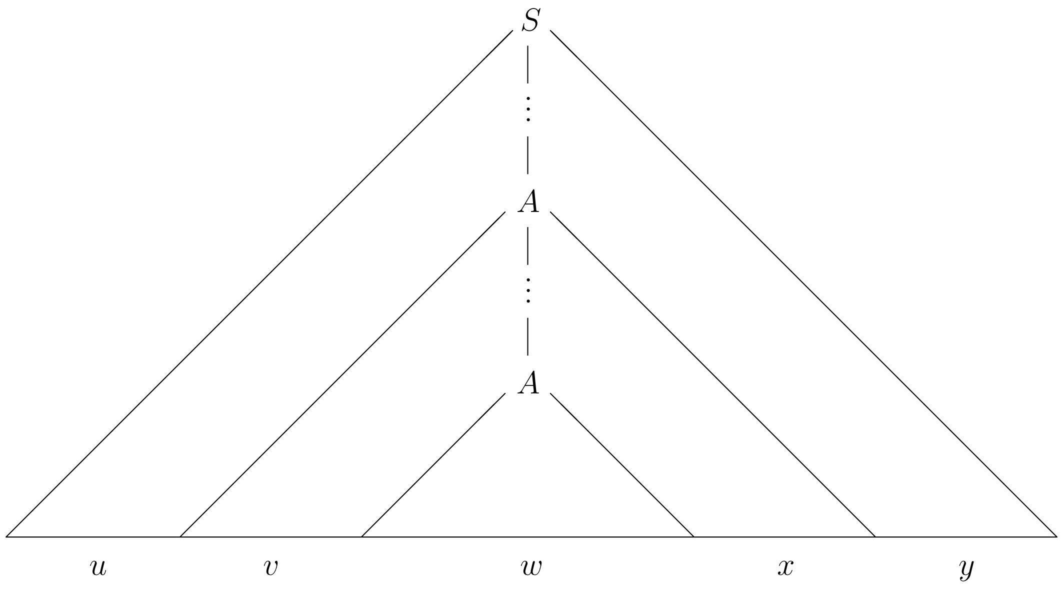


Figure 2.3: Derivation tree with two occurrences of A

Theorem 2.36 *The family $\mathcal{L}(CF)$ is properly included in $\mathcal{L}(CS)$.*

Proof. Since the inclusion is known by (2.7), it is sufficient to show that the inclusion is strict.

We prove that $L = \{a^n b^n c^n \mid n \geq 1\} \in \mathcal{L}(MON) \setminus \mathcal{L}(CF)$. By Example 2.9, $L \in \mathcal{L}(MON)$.

Let us assume that L is a context-free language. By Theorem 2.35, there is a constant k and a partition of $z = a^k b^k c^k = uvwxy$ with the properties ii) and iii) of Theorem 2.35. We discuss only the case that $v \neq \lambda$; the considerations for $v = \lambda$ and $x \neq \lambda$ are analogous and left to the reader.

We distinguish the following cases (by $|vwx| \leq k$, all possible cases are regarded):

Case 1. $v = a^r b^s$ with $r \geq 1, s \geq 0$. By $|vwx| \leq k$, the word vwx contains no occurrence of c. Thus uv^2wx^2y contains at least $k+r > k$ occurrences of the letter a, but only k occurrences of the letter c. Therefore $uv^2wx^2y \notin \{a^n b^n c^n \mid n \geq 1\}$ in contrast to property iii) of Theorem 2.35.

Case 2. $v = b^s c^t$ with $s \geq 1, t \geq 0$. Then vwx contains no occurrence of a, and hence we have $\#_a(uv^2wx^2y) = k$ and $\#_b(uv^2wx^2y) \geq k + s > k$, which leads to a contradiction as in Case 1.

Case 3. $v = c^t$ with $t \geq 1$. Again, vwx contains no occurrence of a, and we get $\#(uv^2wx^2y) = k$ and $\#(uv^2wx^2y) \geq k + t > k$, which leads to a contradiction, again. $\square$

We combine (2.7) and the Theorems 2.32, 2.34, and 16.13 and get immediately the following result.

Theorem 2.37 $\mathcal{L}(REG) \subset \mathcal{L}(LIN) \subset \mathcal{L}(CF) \subset \mathcal{L}(CS) = \mathcal{L}(MON) \subseteq \mathcal{L}(RE)$. $\square$

Thus to obtain a final result it remains to discuss the properness of the inclusion $\mathcal{L}(MON) \subseteq \mathcal{L}(RE)$. This will be done in Chapter **??**.