

Lecture 1

Generative Capacity and Natural Language

Last modified 2016/04/28

Formal grammars

An essential property of human language that distinguishes it from other information-carrying systems (e.g., the system of traffic signs) is its creative aspect: it makes “infinite use of finite means” in the sense that it allows infinitely many ways of combining finitely many basic linguistic units to form arbitrarily complex expressions. (This property is shared by some artificial “languages” like programming languages and languages of symbolic logic.) By a *grammar* we mean a system of rules whereby words—which we assume to be basic linguistic units for the sake of simplicity—may be put together to form sentences. A speaker of a language has a certain grammar internalized in his/her head and uses it to produce and understand sentences of his/her language. What kind of rule systems human grammars are is an empirical issue and a subject matter of linguistics.

One way of approaching this question is to study mathematical models of grammar (*formal grammars*). Different types of formal grammars (or *grammar formalisms*) have different formal properties, which may render them more or less plausible as models of human grammars. Especially, a certain limitation in the expressive power or *generative capacity* of a grammar formalism may be seen to conflict with facts about certain constructions in some existing natural language. Such a discovery would then be considered a reason to reject the grammar formalism in question as a model of human grammar. At the same time, the grammar formalism would serve the useful purpose of characterizing a certain property of natural language in mathematically precise terms: “Natural language goes beyond the expressive power of grammar formalism X.”

There is another way in which the study of grammar formalisms may provide

important insights into human language. A formalization of grammar as an abstract, mathematical object makes it feasible to devise and study various natural algorithms for solving problems like recognition, parsing, etc. (This task is very difficult, if not impossible, when the grammar “formalism” is not precisely defined or is not clearly separated from linguistic hypotheses expressed in terms of it.) The study of algorithmic aspects of grammar formalisms has the potential to provide important tools for the empirical study of human sentence comprehension and production as well as child language acquisition. This is true of research in sentence comprehension, which has been influenced (to a limited extent) by the theory of parsing for context-free grammars.

***n*-gram models**

In an *n*-gram model or $(n - 1)$ -th order Markov process, the probability of an occurrence of a symbol depends on the sequence of $n - 1$ preceding symbols. The statistical structure of English can be approximated by counting the number of *n*-tuples of words of each type in a long sample of text. The following examples, originally from Shannon 1948 and Miller and Selfridge 1950, are cited in Miller and Chomsky 1963. (The “*n*-th order approximation” corresponds to the *n*-gram model.)

- (1.1) First-order approximation (words independent, but with frequencies representative of English): REPRESENTING AND SPEEDILY IS AN GOOD APT OR CAME CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE MESSAGE HAD BE THESE
- (1.2) Second-order word approximation (word-pairs with frequencies representative of English): THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED
- (1.3) Third-order word approximation (word-triplets with frequencies representative of English): FAMILY WAS LARGE DARK ANIMAL CAME ROARING DOWN THE MIDDLE OF MY FRIENDS LOVE BOOKS PASSIONATELY EVERY KISS IS FINE
- (1.4) Fifth-order word approximation (word quintuplets with frequencies representative of English): ROAD IN THE COUNTRY WAS INSANE ESPECIALLY IN DREARY ROOMS WHERE THEY HAVE SOME BOOKS TO BUY FOR STUDYING GREEK

It is not possible to use an n -gram model to characterize the set of grammatical sentences of English. For any finite n , there would be ungrammatical sequences longer than n words that an n -gram model could not reject.

$$(1.5) \quad \text{The } \left\{ \begin{array}{l} \text{mother} \\ \text{parents} \end{array} \right\} \text{ of (the father of)}^k \text{ John } \left\{ \begin{array}{l} \text{is} \\ \text{are} \end{array} \right\} \text{ dead.}$$

For $k > (n-4)/3$, an n -gram model cannot correctly capture the number agreement between *mother/parents* and *is/are* in (1.5).

Figure 1.1 is a *finite automaton* that describes the infinite set of grammatical sentences in (1.5).

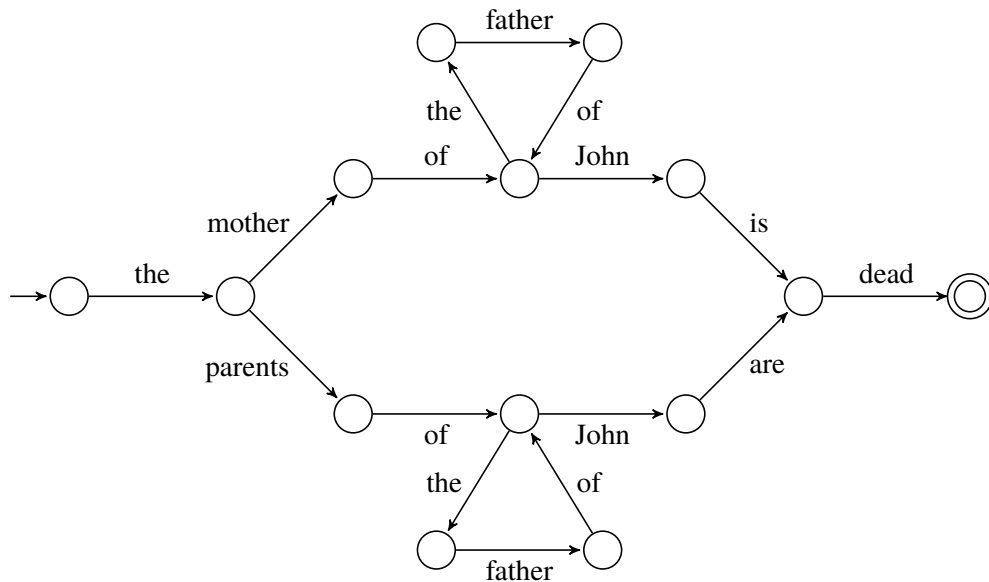


Figure 1.1: A finite automaton accepting the grammatical sentences in (1.5).

A finite automaton has a finite number of *states* and arrows connecting pairs of states. Each arrow is labeled with a *symbol*. In the automaton in Figure 1.1, the labels on the arrows are English words, which are treated as atomic symbols. If you can get from the *initial state* (indicated by an unlabeled arrow pointing to it) to the *final state* (indicated by a double circle) by traversing a sequence of arrows, then the corresponding sequence of symbols (*string*) is *accepted* by the automaton. The automaton in Figure 1.1 accepts an infinite set of strings of English words, all of which are grammatical.

The underlying non-stochastic structure of an n -gram model is a finite automaton of a very restricted kind (an $(n - 1)$ -*limited automaton*) which has a different

state corresponding to each $(n - 1)$ -gram. In an n -limited automaton, the current state is determined by the last n symbols that have been scanned.¹

An n -limited finite automaton cannot capture a dependency between words that are more than n words apart. In natural language, there are *unbounded dependencies* like the one in (1.5), where words that are dependent on each other can occur arbitrarily far apart.

Finite automata and regular languages

Formally, a *deterministic finite automaton* (DFA) is a 5-tuple $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$, where

1. Q is a finite set of *states*,
2. Σ is a finite set called the *input alphabet*,
3. $\delta: Q \times \Sigma \rightarrow Q$ is the *transition function*,²
4. $q_- \in Q$ is the *initial state*, and
5. $F \subseteq Q$ is the set of *final states*.

Let $w = a_1 a_2 \dots a_n$ be a string over the alphabet Σ . Then \mathcal{A} *accepts* w if there is a sequence of states r_0, r_1, \dots, r_n in Q with the following conditions:

1. $r_0 = q_-$,
2. $r_i = \delta(r_{i-1}, a_i)$ for $i = 1, \dots, n$,
3. $r_n \in F$.

Let L be a *language* (i.e., set of strings) over Σ ($L \subseteq \Sigma^*$). We say that \mathcal{A} *recognizes* L if $L = \{ w \in \Sigma^* \mid \mathcal{A} \text{ accepts } w \}$. A language is called *recognizable* or *regular* if it is recognized by some DFA.

Let $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$ be a DFA. It is often useful to define a function $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$ that extends δ as follows:

$$(1.6) \quad \begin{aligned} \hat{\delta}(q, \varepsilon) &= q, \\ \hat{\delta}(q, wa) &= \delta(\hat{\delta}(q, w), a) \quad \text{for } w \in \Sigma^*, a \in \Sigma. \end{aligned}$$

¹The notion of an n -limited automaton appears in Chomsky 1963. It is related to the notion of a *strictly n -testable language* of McNaughton and Papert (1971). See Problem 1.1 at the end of this lecture.

²According to this definition (from Sipser 2012), δ must be a totally defined function. A more common definition of a DFA allows δ to be a partially defined function (as it is in the example in Figure 1.1). To turn a DFA under the more liberal definition into one conforming to the present strict definition, one can add a new “sink” state and define the value of the new transition function to be that state whenever the value of the old transition function is undefined.

It is easy to see that \mathcal{A} accepts w if and only if $\hat{\delta}(q_-, w) \in F$.

Exercise 1.1. Show that for every $w, u \in \Sigma^*$, we have

$$\hat{\delta}(q, wu) = \hat{\delta}(\hat{\delta}(q, w), u).$$

Let $L \subseteq \Sigma^*$. Define a relation $\simeq_L \subseteq \Sigma^* \times \Sigma^*$ by

$$v \simeq_L w \quad \text{if and only if} \quad \{u \mid vu \in L\} = \{u \mid wu \in L\}.$$

Then \simeq_L is an equivalence relation that is *right-invariant* in the sense that

$$v \simeq_L w \quad \text{implies} \quad vu \simeq_L wu \quad \text{for all } v, w, u \in \Sigma^*.$$

The number of equivalence classes of \simeq_L is called the *index* of \simeq_L . The following theorem was originally proved by Myhill (1957) and Nerode (1958):³

Theorem 1.1 (Myhill-Nerode). *L is regular if and only if \simeq_L has finite index.*

Proof. (\Rightarrow). Suppose that L is regular, and L is recognized by a DFA $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$. Let $\hat{\delta}: Q \times \Sigma^* \rightarrow Q$ be the function defined from δ by (1.6).

We claim that $\hat{\delta}(q_-, w) = \hat{\delta}(q_-, v)$ implies $w \simeq_L v$. Since $\hat{\delta}(q_-, x) \in Q$ for all $x \in \Sigma^*$ and Q is finite, it follows from this claim that \simeq_L has finite index.

Suppose $\hat{\delta}(q_-, w) = \hat{\delta}(q_-, v)$. Then, using Exercise 1.1, we have

$$\begin{aligned} \{u \mid wu \in L\} &= \{u \mid \hat{\delta}(q_-, wu) \in F\} \\ &= \{u \mid \hat{\delta}(\hat{\delta}(q_-, w), u) \in F\} \\ &= \{u \mid \hat{\delta}(\hat{\delta}(q_-, v), u) \in F\} \\ &= \{u \mid \hat{\delta}(q_-, vu) \in F\} \\ &= \{u \mid vu \in L\}, \end{aligned}$$

so $w \simeq_L v$. This proves the claim.

(\Leftarrow). Suppose that \simeq_L has finite index, and let $[w_1]_{\simeq_L}, \dots, [w_n]_{\simeq_L}$ be its equivalence classes. (For a string $w \in \Sigma^*$, $[w]_{\simeq_L} = \{x \in \Sigma^* \mid w \simeq_L x\}$.) Define a DFA $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$ as follows:

$$\begin{aligned} Q &= \{[w_1]_{\simeq_L}, \dots, [w_n]_{\simeq_L}\}, \\ \delta([w_i]_{\simeq_L}, a) &= [w_i a]_{\simeq_L} \quad \text{for } i = 1, \dots, n \text{ and } a \in \Sigma, \\ q_- &= [\varepsilon]_{\simeq_L}, \\ F &= \{[w]_{\simeq_L} \mid w \in L\}. \end{aligned}$$

³The relation \simeq_L is sometimes called the *Myhill-Nerode relation*.

Note that the right-invariance of \simeq_L implies that $[v]_{\simeq_L} \in F$ if and only if $v \in L$. Let $\hat{\delta}$ be the function defined from δ by (1.6). We claim that for every $w \in \Sigma^*$, $\hat{\delta}(q_-, w) = [w]_{\simeq_L}$. This implies that \mathcal{A} accepts w if and only if $w \in L$, i.e., \mathcal{A} recognizes L .

We show the claim by induction on the length of w . If $w = \varepsilon$, then $\hat{\delta}(q_-, w) = q_- = [\varepsilon]_{\simeq_L}$, so the claim holds. If $w = w'a$ ($a \in \Sigma$), then $\hat{\delta}(q_-, w'a) = \delta(\hat{\delta}(q_-, w'), a) = \delta([w']_{\simeq_L}, a)$, by induction hypothesis. Let i be the number such that $[w_i]_{\simeq_L} = [w']_{\simeq_L}$. Then $\delta([w']_{\simeq_L}, a) = \delta([w_i]_{\simeq_L}, a) = [w_i a]_{\simeq_L}$. Since $[w_i]_{\simeq_L} = [w']_{\simeq_L}$, $w_i \simeq_L w'$, and by the right-invariance of \simeq_L , $w_i a \simeq_L w' a$. Therefore, $[w_i a]_{\simeq_L} = [w' a]_{\simeq_L}$, and we have shown $\hat{\delta}(q_-, w'a) = [w'a]_{\simeq_L}$. \square

By Theorem 1.1, it is easy to see that $L_1 = \{a^n b^n \mid n \geq 0\}$ is not regular because \simeq_{L_1} has infinite index; for each $n \geq 0$, $\{a^n\}$ is an equivalence class of \simeq_{L_1} .

Exercise 1.2. List all the equivalence classes of \simeq_{L_1} .

By a similar reasoning, we can show that the set of grammatical strings of English words is not regular. Consider the following string of words:

(1.7) people people people see see see

This is a grammatical, albeit awkward and hard to process, sentence of English. It has the following structure:

(1.8) people [people [people see] see] see

The first and second occurrences of *see* are transitive verbs, while the third is an intransitive verb, and the two bracketed substrings are relative clauses modifying the immediately preceding occurrence of *people*. Observe that a string of the form

(1.9) people^m seeⁿ

is grammatical if and only if $m = n$. So *peopleⁱ* and *people^j* belong to different equivalence classes if $i \neq j$. By Theorem 1.1, this means that the set of grammatical strings of English words is not regular.

Even though the acceptability of *peopleⁿ seeⁿ* quickly deteriorates as n grows, the kind of nested structure that (1.8) exemplifies is not uncommon in English (see Hudson 1996 for various examples).

(1.10) A book [that some Italian [I've never heard of] wrote] will be published soon by MIT Press.

(1.11) The policies [that the students [I know] object to most strenuously] are those pertaining to smoking.

(1.12) Anyone₁ who feels that if₂ so-many₃ more₄ students₅ who we₆ haven't₆ actually admitted are₅ sitting in on the course than₄ ones we have that₃ the room had to be changed, then₂ probably auditors will have to be excluded, is₁ likely to agree that the curriculum needs revision.

The example (1.12) is from Chomsky and Miller 1963.

Chomsky (1956) shows English to be non-regular using the following constructions:

- (1.13) a. If S_1 , then S_2
 b. Either S_3 or S_4
 c. The man who said that S_5 is arriving today

Chomsky's argument is not mathematically precise (Daly 1974), but it is easy to use (1.13) to show that English is not regular.

Exercise 1.3. Let $L \subseteq \Sigma^*$. Define a relation $\equiv_L \subseteq \Sigma^* \times \Sigma^*$ by

$$v \equiv_L w \quad \text{if and only if} \quad \{ (x, y) \mid xvy \in L \} = \{ (x, y) \mid xwy \in L \}.$$

Then \equiv_L is a *congruence relation* on Σ^* in the sense that \equiv_L is an equivalence relation and $v \equiv_L v'$ and $w \equiv_L w'$ imply $vw \equiv_L v'w'$. The relation \equiv_L is known as the *syntactic congruence of L* . The set of equivalence classes of \equiv_L (or more precisely, the quotient monoid Σ^*/\equiv_L) is called the *syntactic monoid of L* and is denoted $\text{Syn}(L)$. Show that L is regular if and only if $\text{Syn}(L)$ is finite.

Context-free grammars and context-free languages

The non-regular language $L = \{ a^n b^n \mid n \geq 0 \}$ can be described by a *context-free grammar* with the following set of *productions*:

$$(1.14) \quad \begin{array}{ll} S \rightarrow \varepsilon & A \rightarrow a \\ S \rightarrow ASB & B \rightarrow b \end{array}$$

The string $a^3 b^3$ is *generated* by this grammar with the following *derivation*:

$$\begin{aligned} S &\Rightarrow ASB \Rightarrow aSB \Rightarrow aASBB \Rightarrow aaSBB \Rightarrow aaASBBB \Rightarrow aaaSBBB \\ &\Rightarrow aaaBBB \Rightarrow aaabBB \Rightarrow aaabbB \Rightarrow aaabbb. \end{aligned}$$

Each step of the derivation is sanctioned by one of the productions of the grammar. Formally, a *context-free grammar* (CFG) is a 4-tuple $G = (N, \Sigma, P, S)$, where

1. N is a finite set of symbols called *nonterminals*,

2. Σ is a finite set of symbols called *terminals*, disjoint from N ,
3. P is a finite set of *productions* (or *rules*), with each production being of the form $A \rightarrow \alpha$, where $A \in N$ and $\alpha \in (N \cup \Sigma)^*$, and
4. S is a member of N called the *start symbol*.

For $\beta, \gamma \in (N \cup \Sigma)^*$, we write $\beta \Rightarrow_G \gamma$ if there are a production $A \rightarrow \alpha$ in P and some $\beta_1, \beta_2 \in (N \cup \Sigma)^*$ such that $\beta = \beta_1 A \beta_2$ and $\gamma = \beta_1 \alpha \beta_2$. We say that G *generates* a string $w \in \Sigma^*$ if $S \Rightarrow_G^* w$. (The relation \Rightarrow_G^* is the reflexive transitive closure of the relation \Rightarrow_G .) The set of terminal strings generated by G is the language generated by G and is written $L(G)$. A language is called *context-free* if it is generated by some context-free grammar. All regular languages are context-free, but not vice versa.

Given a CFG G , a *parse tree* (or *derivation tree*) of G is a labeled ordered tree satisfying the following conditions:

1. each non-leaf node is labeled by a nonterminal,
2. each leaf node is labeled by a terminal or ε ,
3. the root node is labeled by the start symbol,
4. for each non-leaf node labeled by a nonterminal A , there is some production $A \rightarrow \alpha$ such that α is the string obtained by reading the labels of its children nodes, from left to right, and
5. any leaf node labeled by ε has no sibling.

The *yield* of a parse tree is the terminal string obtained by reading the labels of the leaf nodes from left to right. An example of a parse tree of the grammar in (1.14) is in Figure 1.2.

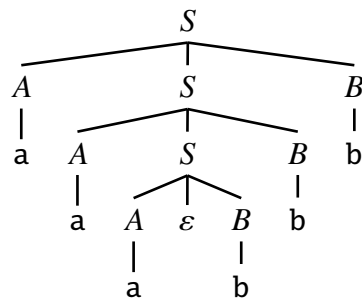


Figure 1.2: A parse tree whose yield is aaabbb.

The language generated by a CFG $G = (N, \Sigma, P, S)$ can be defined alternatively in terms of parse trees:

$$L(G) = \{ w \in \Sigma^* \mid w \text{ is the yield of some parse tree of } G \}.$$

The following grammar generates a small fragment of English. Figure 1.3 contains two parse trees of this grammar.⁴

- | | | | |
|--------|-------------------------|-----------------------------|-------------------------|
| (1.15) | $S \rightarrow NP VP$ | $RC \rightarrow C S/NP$ | $N \rightarrow rat$ |
| | $NP \rightarrow Det N1$ | $S/NP \rightarrow NP VP/NP$ | $N \rightarrow cheese$ |
| | $N1 \rightarrow N$ | $VP/NP \rightarrow Vt$ | $Vt \rightarrow chased$ |
| | $N1 \rightarrow N1 RC$ | $Det \rightarrow the$ | $Vt \rightarrow bit$ |
| | $VP \rightarrow Vt NP$ | $N \rightarrow dog$ | $Vt \rightarrow ate$ |
| | $RC \rightarrow C VP$ | $N \rightarrow cat$ | $C \rightarrow that$ |

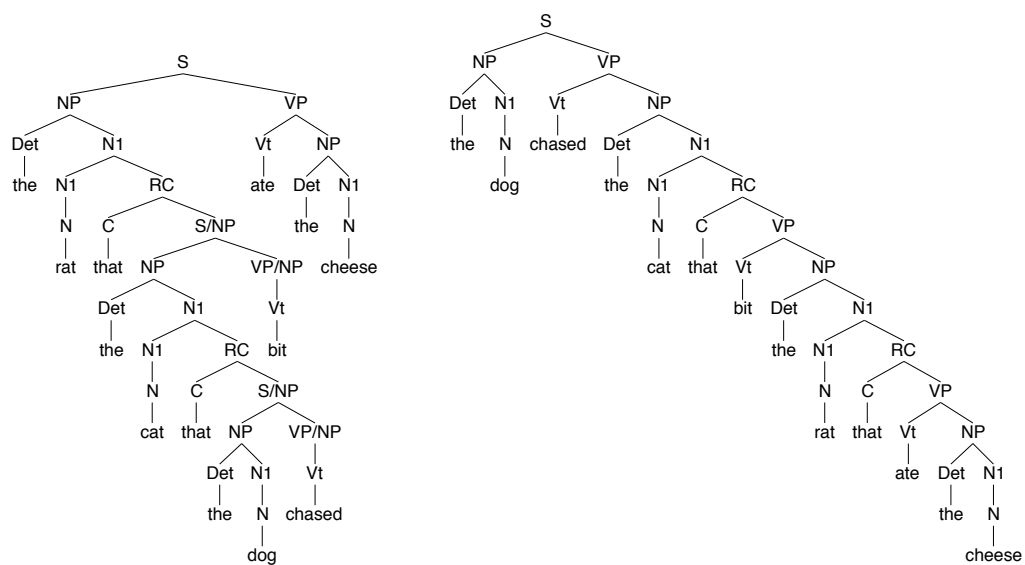


Figure 1.3: Center-embedded and right-branching structures.

⁴The names of nonterminals here are mostly abbreviations of standard grammatical categories, like, Sentence, Noun Phrase, Verb Phrase, etc. Nonterminals of the form X/Y stands for an X that has a Y hole in it. This notation comes from Gazdar et al. 1985.

Exercise 1.4. Consider the following grammar:

(1.16)	$S \rightarrow NP VP$	$RC \rightarrow S/NP$
	$NP \rightarrow N1$	$S/NP \rightarrow NP VP/NP$
	$N1 \rightarrow N$	$VP/NP \rightarrow Vt$
	$VP \rightarrow Vt NP$	$N \rightarrow \text{people}$
	$VP \rightarrow Vi$	$Vt \rightarrow \text{see}$
	$N1 \rightarrow N1 RC$	$Vi \rightarrow \text{see}$

The following are among the strings generated by this grammar. Draw a parse tree for each string.

- people people people see see see
- people people see see people people see
- people people see people see see people people see people see

A CFG G is said to be *self-embedding* if it has a nonterminal A such that $A \Rightarrow_G^* vAw$ for some non-empty strings of terminals v, w . Notice that the grammars in (1.14) and (1.15) are self-embedding.

Theorem 1.2 (Chomsky). *Every context-free grammar that is not self-embedding generates a regular language.*

This theorem was first proved by Chomsky (1959). See also Bar-Hillel et al. 1961, Chomsky 1963, Nederhof 2000, and Problems 2.2 and 2.3 at the end of Lecture 2.



Self-embedded structures like those in (1.10) and (1.11) are abundant in natural language. It is much harder to find evidence that natural language goes beyond the expressive power of context-free grammars (see Pullum and Gazdar 1982 for the history of failed attempts). *Cross-serial dependencies* found in the Zürich dialect of Swiss German provide one such piece of evidence (Huybregts 1984, Shieber 1985).

A crucial fact about Swiss German is that like standard German, it marks verb objects with different case (dative or accusative) depending on the type of the verb. The following examples are from Shieber 1985. They are all subordinate clauses and should be envisaged as preceded by the string “de Jan säit, dass” (“Jan says that”) or something similar to form a complete sentence. The asterisk * in front of an example indicates ungrammaticality.

(1.17)	mer em Hans	es huus	hälfed	aastriiche
	we	Hans-DAT	the house-ACC	helped paint
	‘we helped Hans paint the house’			

- (1.18) mer de Hans es huus lönd aastriiche
 we Hans-ACC the house-ACC let paint
 ‘we let Hans paint the house’
- (1.19) *mer em Hans es huus lönd aastriiche
 we Hans-DAT the house-ACC let paint
 ‘we let Hans paint the house’
- (1.20) *mer de Hans em huus lönd aastriiche
 we Hans-ACC the house-DAT let paint
 ‘we let Hans paint the house’
- (1.21) mer d’chind em Hans es huus lönd hälfe aastriiche
 we the children-ACC Hans-DAT the house-ACC let help paint
 ‘we let the children help Hans paint the house’
- (1.22) *mer d’chind de Hans es huus lönd hälfe aastriiche
 we the children-ACC Hans-ACC the house-ACC let help paint
 ‘we let the children help Hans paint the house’
- (1.23) mer em Hans es huus haend wele hälfe aastriiche
 we Hans-DAT the house-ACC have wanted help paint
 ‘we have wanted to help Hans paint the house’
- (1.24) mer d’chind em Hans es huus haend wele laa hälfe
 we the children-ACC Hans-DAT the houes-ACC have wanted let help
 aastriiche
 paint
 ‘we have wanted to let the children help Hans paint the house’

Notice that in all these examples, the object noun phrases all precede the verbs, and the order in which the verbs appear corresponds to the order in which their objects appear. This configuration is called “cross-serial”, because if we draw lines connecting corresponding pairs of noun phrases and verbs, the lines do not nest but rather cross each other.

- (1.17) mer em Hans es huus hölfed aastriiche

 we Hans-DAT the house-ACC let paint
- (1.21) mer d’chind em Hans es huus lönd hälfe aastriiche

 we the children-ACC Hans-DAT the house-ACC let help paint

This is by no means the only possible word order in this construction; Swiss German allows nested configurations as well, just like standard German, and it is also possible to have an object noun phrase follow a verb (but not the one whose object it is). The most important thing here is that the cross-serial word order seems to be generally available.

The following sentence is grammatical if and only if $k = m$ and $l = n$.

(1.25) De Jan säit, dass mer (d'chind)^k (em Hans)^l es huus haend wele laa^m hälfeⁿ aasriiche.

The following theorem was first proved by Bar-Hillel et al. (1961) and is known as the *pumping lemma* (see, e.g., Sipser 2012 for proof):

Theorem 1.3 (Bar-Hillel, Perles, and Shamir). *Let L be a context-free language. Then there is a natural number p such that for every string $z \in L$ with $|z| \geq p$, there are strings u, v, w, x, y satisfying the following conditions:*

1. $z = uvwxy$,
2. $|vx| \geq 1$,
3. $|vwx| \leq p$, and
4. $uv^nwx^ny \in L$ for all $n \geq 0$.

Exercise 1.5. Use the pumping lemma to show that the following languages are non-context-free:

- $\{xx \mid x \in \{a, b\}^*\}$
- $\{a^n b^n c^n \mid n \geq 0\}$
- $\{a^m b^n a^m b^n \mid m, n \geq 0\}$

The following theorems say that the class of context-free languages is closed under homomorphism and intersection with regular sets. With the help of these theorems, we can show that the set of grammatical word strings of Swiss German is not context-free.

A mapping h from Σ_1^* to Σ_2^* is a *homomorphism* if $h(xy) = h(x)h(y)$ for all $x, y \in \Sigma_1^*$. A homomorphism is completely determined by the values it takes on strings of length 1 (i.e., symbols).

Theorem 1.4. *Let $L \subseteq \Sigma_1^*$ be a context-free language. If $h: \Sigma_1^* \rightarrow \Sigma_2^*$ is a homomorphism, then $h(L) = \{h(w) \mid w \in L\}$ is a context-free language.*

Proof. Let $G = (N, \Sigma_1, P, S)$ be a CFG generating L . Extend h to a function from $(N \cup \Sigma_1)^* \rightarrow (N \cup \Sigma_2)^*$ by setting $h(A) = A$ for all $A \in N$. Let

$$P' = \{A \rightarrow h(\alpha) \mid A \rightarrow \alpha \in P\}.$$

Then it is easy to show that $G' = (N, \Sigma_2, P', S)$ generates $h(L)$. □

Theorem 1.5. *If L is a context-free language and R a regular language, then $L \cap R$ is a context-free language.*

Proof. Let $G = (N, \Sigma, P, S)$ be a CFG generating L and $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$ be a DFA recognizing R . Define a CFG $G' = (N' \cup \{S'\}, \Sigma', P', S')$ by

$$\begin{aligned} N' &= \{ A^{(q,r)} \mid A \in N \text{ and } q, r \in Q \}, \\ \Sigma' &= \{ a^{(q,r)} \mid a \in \Sigma, q, r \in Q, \text{ and } \delta(q, a) = r \}, \\ P' &= \{ S' \rightarrow S^{(q_-,r)} \mid r \in F \} \cup \\ &\quad \{ A^{(q,r)} \rightarrow X_1^{(r_0,r_1)} \dots X_n^{(r_{n-1},r_n)} \mid A \rightarrow X_1 \dots X_n \in P, q = r_0, r = r_n, \text{ and} \\ &\quad X_i^{(r_{i-1},r_i)} \in N' \cup \Sigma' \text{ for } i = 1, \dots, n \}. \end{aligned}$$

Then we can show

$$L(G') = \{ a_1^{(r_0,r_1)} \dots a_n^{(r_{n-1},r_n)} \mid a_1 \dots a_n \in L, r_0 = q_-, r_n \in F, \text{ and } \delta(r_{i-1}, a_i) = r_i \text{ for } i = 1, \dots, n \}.$$

Let $h: \Sigma'^* \rightarrow \Sigma^*$ be the homomorphism such that $h(a^{(q,r)}) = a$ for all $a^{(q,r)} \in \Sigma'$. Then $h(L(G')) = L \cap R$, which is a context-free language by Theorem 1.4. \square

Exercise 1.6. Use Theorems 1.3, 1.4, and 1.5 to show that the set of grammatical word strings of Swiss German is not context-free.

A class \mathcal{C} of languages is called a *rational cone* if it is closed under homomorphism, inverse homomorphism (i.e., $L \in \mathcal{C}$ implies $h^{-1}(L) = \{ w \mid h(w) \in L \} \in \mathcal{C}$ for any homomorphism h), and intersection with regular sets. The regular languages and the context-free languages each constitute a rational cone. Most important classes of languages we will consider are rational cones.

Problems

1.1. A DFA $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$ is *n-limited* if and only if it satisfies the following conditions:

- There is a state $q_\infty \in Q - F$ such that $\delta(q_\infty, a) = q_\infty$ for every $a \in \Sigma$; and
- For all $q, q' \in Q$ and for all $x \in \Sigma^n$, $\hat{\delta}(q, x) \neq q_\infty$ and $\hat{\delta}(q', x) \neq q_\infty$ implies $\hat{\delta}(q, x) = \hat{\delta}(q', x)$.

Let $x \in \Sigma^n \Sigma^*$. Define

$$\begin{aligned} \alpha_n(x) &= \text{the prefix of } x \text{ of length } n, \\ \omega_n(x) &= \text{the suffix of } x \text{ of length } n, \end{aligned}$$

$$\beta_n(x) = \{ y \in \Sigma^n \mid u y v = x \text{ for some } u, v \in \Sigma^+ \}.$$

A language $L \in \Sigma^*$ is *strictly n -testable* if there exist subsets α, ω, β of Σ^n such that for every $x \in \Sigma^n \Sigma^*$, $x \in L$ if and only if $\alpha_n(x) \in \alpha$, $\omega_n(x) \in \omega$, and $\beta_n(x) \subseteq \beta$.

- (a) Let $\#$ be a symbol not in Σ . Show that for every $L \in \Sigma^*$, $L = L(\mathcal{A})$ for some n -limited DFA \mathcal{A} if and only if $\#L\#$ is strictly $(n + 1)$ -testable.
- (b) Show that the class of strictly n -testable languages is closed under intersection, but not union or complementation.
- (c) Show that every regular language is a homomorphic image of a strictly 2-testable language.

1.2. The “if” direction of Theorem 1.1 constructed a DFA out of a language L that has finite index.

- (a) Show that the DFA constructed in the “if” direction of Theorem 1.1 is a minimal DFA (DFA with the smallest number of states) recognizing L .
- (b) Show that up to isomorphism, each regular language has a unique minimal DFA recognizing it.

1.3. An equivalence relation \simeq on Σ^* is *right-invariant* if for all $x, y, u \in \Sigma^*$, $x \simeq y$ implies $xu \simeq yu$. A state q of a DFA $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$ is *reachable* if there is some $x \in \Sigma^*$ such that $\hat{\delta}(q_-, x) = q$. Show that up to isomorphism, DFAs whose input alphabet is Σ and whose states are all reachable correspond one-to-one with the pairs (\simeq, E) such that \simeq is a right-invariant equivalence relation on Σ^* of finite index and E is a subset of its equivalence classes.

1.4. A *monoid* is a set M equipped with an associative binary operation $\circ: M \times M \rightarrow M$ and an identity $1 \in M$ satisfying $1 \circ x = x \circ 1 = x$ for all $x \in M$. The set Σ^* of all finite strings over an alphabet Σ is a monoid, with concatenation as monoid operation and ε as identity. (This is called the *free monoid* generated by Σ .) If $(M, \circ_M, 1_M)$ and $(N, \circ_N, 1_N)$ are monoids, a mapping $h: M \rightarrow N$ is a *monoid homomorphism* if $h(1_M) = 1_N$ and $h(x \circ_M y) = h(x) \circ_N h(y)$ for all $x, y \in M$.

A finite monoid M *recognizes* a language $L \subseteq \Sigma^*$ if there is a monoid homomorphism $h: \Sigma^* \rightarrow M$ and a subset M' of M such that $L = h^{-1}(M')$. Prove that $L \subseteq \Sigma^*$ is recognized by some finite monoid if and only if it is a regular language.

1.5. If E is a set, a *transformation monoid* on E is a monoid $(M, \circ, 1)$, where M is a set of functions from E to E , \circ is the composition operation, and 1 is the identity function on M . The *transition monoid* of a DFA $\mathcal{A} = (Q, \Sigma, \delta, q_-, F)$ is the transformation monoid generated by $\{ [a] \mid a \in \Sigma \}$, where $[a]$ is the function from Q to Q given by $[a](q) = \delta(q, a)$.

- (a) Show that if L is recognized by \mathcal{A} , then L is recognized by its transition monoid.
- (b) Show that if \mathcal{A} is a minimal DFA for L , then the transition monoid of \mathcal{A} is isomorphic to the syntactic monoid of L .

(See Propositions 3.18 and 4.27 of Pin (in preparation).)

1.6. A CFG is *ambiguous* if it generates some string with more than one parse tree.

- (a) Show that the grammar in (1.15) is ambiguous.
- (b) Write an unambiguous grammar that generates the same language as the grammar in (1.15).

1.7. Consider the grammar (1.16) given in Exercise 1.4. Show that the set

$$\{ w \mid \text{NP} \Rightarrow^* \text{people } w \}$$

coincides with the *Dyck language* over $\{ \text{people, see} \}$, i.e., the language defined by the following context-free grammar:⁵

$$S \rightarrow \varepsilon \mid \text{people } S \text{ see } S.$$

1.8. See Problem 1.4 for the definitions of a monoid and of recognition by a finite monoid.

1. Let $(M, \circ_M, 1_M)$ be a finite monoid, and let $h: \Sigma^* \rightarrow M$ be a monoid homomorphism. Prove that for every CFG $G = (N, \Sigma, P, S)$, there is a CFG $G' = (N', \Sigma, P', S')$ that satisfies the following condition:
 - (a) $L(G) = L(G')$, and
 - (b) for every $A \in N' - \{S'\}$ and $w, w' \in \Sigma^*$, $A \Rightarrow_{G'}^* w$ and $A \Rightarrow_{G'}^* w'$ imply $h(w) = h(w')$.
2. Give an alternative proof of Theorem 1.5 using a finite monoid recognizing R instead of a DFA for R .

1.9. Prove the following weak form of *Ogden's lemma* for context-free grammars (Ogden 1968):

Let $G = (N, \Sigma, P, S)$ be a CFG. There is a natural number p such that for every string $z \in L(G)$ and $J \subseteq [1, |z|]$, if $|J| \geq p$, then z can be written as $z = uvwxy$ so that

⁵Vertical bars are often used to combine multiple productions that share the same left-hand side. Thus, $A \rightarrow \alpha \mid \beta$ abbreviates two productions $A \rightarrow \alpha$ and $A \rightarrow \beta$.

1. at least one of the following sets is non-empty:

$$J \cap [|u| + 1, |uv|],$$

$$J \cap [|uvw| + 1, |uvw x|].$$

2. for some $A \in N$, we have

$$S \Rightarrow_G^* uAy,$$

$$A \Rightarrow_G^* vAx,$$

$$A \Rightarrow_G^* w.$$

(As a consequence, $uv^nwx^n \in L(G)$ for all $n \geq 0$.)

1.10. Show that $L = \{a^m b^m c^n \mid m, n \geq 0\} \cup \{a^m b^n c^n \mid m, n \geq 0\}$ is an *inherently ambiguous* context-free language. In particular, every CFG G such that $L(G) = L$ allows two distinct parse trees for some string of the form $a^n b^n c^n$.

1.11. We refer to the number of nodes in a parse tree T as the *size* of T . If G is a CFG and $z \in L(G)$, we write $dc_G(z)$ for the size of the smallest parse tree of G whose yield is z .

- (a) Let G be a CFG without ε -productions. Show that for every $z \in L(G)$, it holds that $dc_G(z) \leq (3n + 1) \cdot |z| - n$, where n is the number of nonterminals of G .
- (b) Show that for every CFG G , there exist numbers a, b (dependent on G) such that $dc_G(z) \leq a|z| + b$ for every $z \in L(G)$.
- (c) How do a and b depend on G ? (The pair (a, b) is known as the *derivational complexity* of G (Sippu 1982).)

1.12. Consider the following one-person game. You are given a finite number of boxes, each of which can contain, at any point in time, at most one string (of arbitrary length) over an alphabet Σ . One of the boxes, call it S , is designated as the goal box. You are also given a finite number of instructions of the form:

$$A(\alpha) \leftarrow B_1(x_1), \dots, B_n(x_n),$$

where $n \in \mathbb{N}$, A, B_1, \dots, B_n are names of boxes, x_1, \dots, x_n are pairwise distinct variables, and α is a string in $\Sigma^* x_1 \Sigma^* \dots \Sigma^* x_n \Sigma^*$. The interpretation of the rule is as follows. When boxes B_1, \dots, B_n hold strings w_1, \dots, w_n , respectively, you are allowed to put the string $\alpha[x_1 := w_1, \dots, x_n := w_n]$ into box A , replacing any string previously in A . The special case of $n = 0$ means that you are always allowed to throw $\alpha \in \Sigma^*$ into A (discarding any string already in A). You start the

game with each box being empty and a particular goal string $w \in \Sigma^*$ singled out. When you reach a situation where w is in the goal box S , you win the game.

Suppose that the boxes and the instructions are fixed. Show that the set

$$\{ w \mid \text{you can win the game with } w \text{ as goal string} \}$$

is a context-free language. *Hint:* See Ginsburg and Spanier 1968.

1.13. We say that a language $L \subseteq \Sigma^*$ has the *shrinking property* if for any positive integer m , there exists a $k > m$ such that every string $w \in L$ with $|w| \geq k$ can be written as $w = w_1 \dots w_r$ in such a way that the following conditions hold:

- $m < r \leq k$,
- $w_i \neq \varepsilon$ for $i = 1, \dots, r$, and
- every choice of m elements from $\{1, \dots, r\}$ is included in some $\{i_1, i_2, \dots, i_t\}$ such that $t < r$, $1 \leq i_1 < i_2 < \dots < i_t \leq r$, and $w_{i_1} w_{i_2} \dots w_{i_t} \in L$.

(a) Show that every regular language has the shrinking property. (*Hint:* Let $k = (m + 1)|Q|$, where Q is the set of states of a DFA recognizing L .)

(b) Show that every context-free language has the shrinking property.

(Gilman (1996) proves that every indexed language has the shrinking property.)

References

Bar-Hillel, Y., M. Perles, and E. Shamir. 1961. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung* 14(2):143–172.

Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* 2(3):113–124.

Chomsky, Noam. 1959. On certain formal properties of grammars. *Information and Control* 2(2):137–167.

Chomsky, Noam. 1963. Formal properties of grammars. In R. D. Luce, R. R. Bush, and E. Galanter, eds., *Handbook of Mathematical Psychology, Volume II*, pages 323–418. New York: John Wiley and Sons.

Chomsky, Noam and George A. Miller. 1963. Introduction to the formal analysis of natural languages. In R. D. Luce, R. R. Bush, and E. Galanter, eds., *Handbook of Mathematical Psychology, Volume II*, pages 269–321. New York: John Wiley and Sons.

- Daly, Richard Timon. 1974. *Application of the Mathematical Theory of Linguistics*. The Hague: Mouton.
- Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalized Phrase Structure Grammar*. Oxford: Blackwell.
- Gilman, Robert H. 1996. A shrinking lemma for indexed languages. *Theoretical Computer Science* 163:277–281.
- Ginsburg, Seymour and Edwin H. Spanier. 1968. Derivation-bounded languages. *Journal of Computer and System Sciences* 2:228–250.
- Hudson, Richard. 1996. The difficulty of (so-called) self-embedded structures. In P. Backley and J. Harris, eds., *UCL Working Papers in Linguistics* 8, pages 283–314. University College London Department of Phonetics and Linguistics.
- Huybregts, Riny. 1984. The weak inadequacy of context free phrase structure grammars. In G. J. de Haan, M. Trommelen, and W. Zonneveld, eds., *Van Periferie naar Kern*, pages 81–90. Dordrecht: Foris Publications.
- McNaughton, Robert and Seymour A. Papert. 1971. *Counter-Free Automata*. Cambridge, Mass.: MIT Press.
- Miller, George A. and Noam Chomsky. 1963. Finitary models of language users. In R. D. Luce, R. R. Bush, and E. Galanter, eds., *Handbook of Mathematical Psychology, Volume II*, pages 419–491. New York: John Wiley and Sons.
- Myhill, John. 1957. Finite automata and the representation of events. *Wright Air Development Command Technical Report* 57–624:112–137.
- Nederhof, Mark-Jan. 2000. Practical experiments with regular approximation of context-free languages. *Computational Linguistics* 26(1):17–44.
- Nerode, Anil. 1958. Linear automaton transformations. *Proceedings of the American Mathematical Society* 9(4):541–544.
- Ogden, William. 1968. A helpful result for proving inherent ambiguity. *Mathematical Systems Theory* 2(3):191–194.
- Pin, Jean-Éric. in prepration. Mathematical foundations of automata theory. Book draft available at <https://www.irif.univ-paris-diderot.fr/~jep/PDF/MPRI/MPRI.pdf>.
- Pullum, Geoffrey K. and Gerald Gazdar. 1982. Natural languages and context-free languages. *Linguistics and Philosophy* 4(4):471–504.

- Shieber, Stuart M. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8(3):333–343.
- Sippu, Seppo. 1982. Derivational complexity of context-free grammars. *Information and Control* 53(1–2):52–65.
- Sipser, Michael. 2012. *Introduction to the Theory of Computation, Third Edition*. Boston: Cengage Learning.