Chapter 8

Phrase-Structure Grammars and Context-Sensitive Grammars

8.1 Phrase-Structure Grammars

Context-free grammars can be generalized in various ways. The most general grammars generate exactly the recursively enumerable languages.

Between the context-free languages and the recursively enumerable languages, there is a natural class of languages, the context-sensitive languages.

The context-sensitive languages also have a Turing-machine characterization. We begin with phrase-structure gammars.

Definition 8.1.1 A *phrase-structure grammar* is a quadruple $G = (V, \Sigma, P, S)$, where

- V is a finite set of symbols called the *vocabulary (or set of grammar symbols)*;
- $\Sigma \subseteq V$ is the set of *terminal symbols (for short, terminals)*;
- $S \in (V - \Sigma)$ is a designated symbol called the *start symbol*;

  The set $N = V - \Sigma$ is called the set of *nonterminal symbols (for short, nonterminals).*
- $P \subseteq V^*NV^* \times V^*$ is a finite set of *productions (or rewrite rules, or rules).*

Every production $\langle \alpha, \beta \rangle$ is also denoted as $\alpha \to \beta$. A production of the form $\alpha \to \epsilon$ is called an *epsilon rule, or null rule*.

Example 1.

$$G_1 = (\{S, A, B, C, D, E, a, b\}, \{a, b\}, P, S),$$

where P is the set of rules

$$\begin{aligned}
S &\longrightarrow ABC, \\
AB &\longrightarrow aAD, \\
AB &\longrightarrow bAE, \\
DC &\longrightarrow BaC, \\
EC &\longrightarrow BbC, \\
Da &\longrightarrow aD, \\
Db &\longrightarrow bD, \\
Ea &\longrightarrow aE, \\
Eb &\longrightarrow bE, \\
AB &\longrightarrow \epsilon, \\
C &\longrightarrow \epsilon, \\
aB &\longrightarrow Ba, \\
bB &\longrightarrow Bb.
\end{aligned}$$

It can be shown that this grammar generates the language

$$L = \{ww \mid w \in \{a, b\}^*\},$$

which is not context-free.

8.2 Derivations and Type-0 Languages

The productions of a grammar are used to derive strings. In this process, the productions are used as rewrite rules.

Definition 8.2.1 Given a phrase-structure grammar $G = (V, \Sigma, P, S)$, the (one-step) *derivation relation* $\Longrightarrow_G$ *associated with* G is the binary relation $\Longrightarrow_G \subseteq V^* \times V^*$ defined as follows: for all $\alpha, \beta \in V^*$, we have

$$\alpha \Longrightarrow_G \beta$$

iff there exist $\lambda, \rho \in V^*$, and some production $(\gamma \to \delta) \in P$, such that

$$\alpha = \lambda\gamma\rho \quad \text{and} \quad \beta = \lambda\delta\rho.$$

The transitive closure of $\Longrightarrow_G$ is denoted as $\stackrel{+}{\Longrightarrow}_G$ and the reflexive and transitive closure of $\Longrightarrow_G$ is denoted as $\stackrel{*}{\Longrightarrow}_G$.

When the grammar G is clear from the context, we ususally omit the subscript G in $\Longrightarrow_G$, $\stackrel{+}{\Longrightarrow}_G$, and $\stackrel{*}{\Longrightarrow}_G$.

The language generated by a phrase-structure grammar is defined as follows.

Definition 8.2.2 Given a phrase-structure grammar $G = (V, \Sigma, P, S)$, the *language generated by* G is the set

$$L(G) = \{w \in \Sigma^* \mid S \stackrel{+}{\Longrightarrow} w\}.$$

A language $L \subseteq \Sigma^*$ is a *type*-0 *language* iff $L = L(G)$ for some phrase-structure grammar G.

The following lemma can be shown.

Lemma 8.2.3 *A language L is recursively enumerable iff it generated by some phrase-structure grammar G.*

In one direction, we can construct a nondeterministic Turing machine simulating the derivations of the grammar G. In the other direction, we construct a grammar simulating the computations of a Turing machine.

We now consider some variants of the phrase-structure grammars.

8.3 Type-0 Grammars and Context-Sensitive Grammars

We begin with type-0 grammars. At first glance, it may appear that they are more restrictive than phrase-structure grammars, but this is not so.

Definition 8.3.1 A *type*-0 *grammar* is a phrase-structure grammar $G = (V, \Sigma, P, S)$, such that the productions are of the form

$$\alpha \rightarrow \beta,$$

where $\alpha \in N^+$. A production of the form $\alpha \rightarrow \epsilon$ is called an *epsilon rule, or null rule*.

Lemma 8.3.2 *A language L is generated by a phrase-structure grammar iff it is generated by some type-*0 *grammar.*

We now place additional restrictions on productions, obtaining context-sensitive grammars.

Definition 8.3.3 A *context-sensitive grammar (for short, csg)* is a phrase-structure grammar $G = (V, \Sigma, P, S)$, such that the productions are of the form

$$\alpha A\beta \rightarrow \alpha\gamma\beta,$$

with $A \in N$, $\gamma \in V^+$, $\alpha, \beta \in V^*$, or

$$S \rightarrow \epsilon,$$

and if $S \rightarrow \epsilon \in P$, then S does not appear on the right-hand side of any production.

The notion of derivation is defined as before. A language L is *context-sensitive* iff it is generated by some context-sensitive grammar.

We can also define monotonic grammars.

Definition 8.3.4 A *monotonic grammar* is a phrase-structure grammar $G = (V, \Sigma, P, S)$, such that the productions are of the form

$$\alpha \rightarrow \beta$$

with $\alpha, \beta \in V^+$ and $|\alpha| \leq |\beta|$, or

$$S \rightarrow \epsilon,$$

and if $S \rightarrow \epsilon \in P$, then S does not appear on the right-hand side of any production.

Example 2.

$$G_2 = (\{S, A, B, C, a, b, c\}, \{a, b, c\}, P, S),$$

where P is the set of rules

$$\begin{aligned}
S &\longrightarrow ABC,\\
S &\longrightarrow ABCS,\\
AB &\longrightarrow BA,\\
AC &\longrightarrow CA,\\
BC &\longrightarrow CB,\\
BA &\longrightarrow AB,\\
CA &\longrightarrow AC,\\
CB &\longrightarrow BC,\\
A &\longrightarrow a,\\
B &\longrightarrow b,\\
C &\longrightarrow c.
\end{aligned}$$

It can be shown that this grammar generates the language

$$L = \{w \in \{a, b, c\}^+ \mid \#(a) = \#(b) = \#(c)\},$$

which is not context-free.

Lemma 8.3.5 *A language L is generated by a context-sensitive grammar iff it is generated by some monotonic grammar.*

Lemma 8.3.5 is proved as follows:

Proof.

Step 1. Construct a new monotonic grammar G_1 such that the rules are of the form

$$\alpha \rightarrow \beta,$$

with $|\alpha| \leq |\beta|$ and $\alpha \in N^+$, or $S \rightarrow \epsilon$, where S does not appear on the left-hand side of any rule.

This can be achieved by replacing every terminal a occurring on the left hand-side of a rule by a new nonterminal X_a and adding the rule

$$X_a \rightarrow a.$$

Step 2. Given a rule $\alpha \to \beta$, let

$$w(G) = \max\{|\beta| \mid \alpha \to \beta \in G\}.$$

Construct a new monotonic grammar G_2 such that the rules $\alpha \to \beta$ satisfy the conditions:

(1) $\alpha \in N^+$

(2) $w(G_2) \leq 2$.

Given a rule

$$\pi\colon A_1 \cdots A_m \to B_1 \cdots B_n,$$

with $m \leq n$,

if $n \leq 2$, OK;

If $2 \leq m < n$, create the two rules

$$\begin{aligned} A_1 \cdots A_m &\to B_1 \cdots B_{m-1} X_\pi, \\ X_\pi &\to B_m \cdots B_n. \end{aligned}$$

If $m = 1$ and $n \geq 3$, create the $n - 1$ rules:

$$\begin{aligned}
A_1 &\to B_1 X_{\pi,1}, \\
X_{\pi,1} &\to B_2 X_{\pi,2}, \\
\cdots &\to \cdots, \\
X_{\pi,n-2} &\to B_{n-1} B_n.
\end{aligned}$$

If $m = n$ and $n \geq 3$, create the $n - 1$ rules:

$$\begin{aligned}
A_1 A_2 &\to B_1 X_{\pi,1}, \\
X_{\pi,1} A_3 &\to B_2 X_{\pi,2}, \\
\cdots &\to \cdots, \\
X_{\pi,n-2} A_n &\to B_{n-1} B_n.
\end{aligned}$$

In all cases, $w(G_2)$ is reduced.

Step 3. Create a context-sensitive grammar as follows:

If $A \rightarrow \beta$, OK

If $AB \rightarrow CD$ and $A = C$ or $D = B$, OK

If $\pi\colon AB \rightarrow CD$, where $A \neq C$ and $D \neq B$, create the four rules

$$\begin{aligned} AB &\rightarrow [\pi, A]B, \\ [\pi, A]B &\rightarrow [\pi, A][\pi, B], \\ [\pi, A][\pi, B] &\rightarrow C[\pi, B], \\ C[\pi, B] &\rightarrow CD. \end{aligned}$$

Context-sensitive languages are recursive. This is shown as follows. For any $n \geq 1$ define the sequence of sets $W_i^n \subseteq V^+$, as follows:

$$\begin{aligned} W_0^n &= \{S\}, \\ W_{i+1}^n &= W_i^n \cup \{\beta \in V^+ \mid \alpha \Longrightarrow \beta,\, \alpha \in W_i^n,\, |\beta| \leq n\}. \end{aligned}$$

It is clear that

$$W_0^n \subseteq W_1^n \subseteq \cdots \subseteq W_i^n \subseteq W_{i+1}^n \subseteq \cdots,$$

and if $|V| = K$, since V^i contains K^i strings and since

$$W_i^n \subseteq \bigcup_{j=1}^{n} V^j,$$

every W_i^n contains at most $K + K^2 + \cdots + K^n$ strings, and by the familiar argument, there is some smallest i, say i_0, such that

$$W_{i_0}^n = W_{i_0+1}^n,$$

and $W_j^n = W_{i_0}^n$ for all $j > i_0$.

The following lemma holds.

Lemma 8.3.6 *Given a context-sensitive grammar G, for every $n \geq 1$, for every $i \geq 0$,*

$$W_i^n = \{\beta \in V^+ \mid S \stackrel{k}{\Longrightarrow} \beta,\ k \leq i,\ |\beta| \leq n\}.$$

Furthermore, there is some smallest i, say i_0 such that

$$W_{i_0}^n = \{\beta \in V^+ \mid S \stackrel{*}{\Longrightarrow} \beta,\ |\beta| \leq n\}.$$

Proof. By definition of W_i^n, it is obvious that

$$W_i^n \subseteq \{\beta \in V^+ \mid S \stackrel{k}{\Longrightarrow} \beta,\ k \leq i,\ |\beta| \leq n\}.$$

Conversely, to show that

$$\{\beta \in V^+ \mid S \stackrel{k}{\Longrightarrow} \beta,\ k \leq i,\ |\beta| \leq n\} \subseteq W_i^n,$$

we proceed by induction on i.

The claim is trivial for $i = 0$. Given a derivation

$$S \stackrel{k}{\Longrightarrow} \delta \Longrightarrow \beta,\ k \leq i,\ |\beta| \leq n,$$

we must have $|\delta| \leq n$, since otherwise, because the grammar is context-sensitive, we must have $|\delta| \leq |\beta|$, and we would have $|\beta| > n$, a contradiction.

By the induction hypothesis, we get $\delta \in W_i^n$, and by the definition of W_{i+1}^n, we have $\beta \in W_{i+1}^n$.

For the second part of the lemma, if $|\beta| = n$ with $n \geq 1$, there is some $k \geq 0$ such that $S \stackrel{k}{\Longrightarrow} \beta$.

But then, $\beta \in W_k^n$, which implies that $\beta \in W_{i_0}^n$, since

$$W_0^n \subseteq W_1^n \subseteq \cdots \subseteq W_{i_0}^n,$$

and $W_j^n = W_{i_0}^n$ for all $j > i_0$.

As a corollary of lemma 8.3.6, given any $\beta \in V^*$, we can decide whether $S \stackrel{*}{\Longrightarrow} \beta$.

Indeed, if $\beta = \epsilon$, we must have the production $S \longrightarrow \epsilon$.

Otherwise, if $|\,\beta\,| = n$ with $n \geq 1$, by lemma 8.3.6, we have $\beta \in W_{i_0}^n$.

Thus, is is enough to compute $W_{i_0}^n$ and to test whether β is in it. □

Remark: If the grammar G is **not** context-sensitive, we can't claim that

$$W_i^n = \{\beta \in V^+ \mid S \stackrel{k}{\Longrightarrow} \beta,\, k \leq i,\, |\beta| \leq n\},$$

but the other facts remain true. Unfortunately, $W_{i_0}^n$ may not be computable any more!

The context-sensitive languages are accepted by space-bounded Turing machines, defined as follows.

Definition 8.3.7 A *linear-bounded automaton (for short, lba)* is a nondeterministic Turing machine such that for every input $w \in \Sigma^*$, there is some accepting computation in which the tape contains at most $|w| + 1$ symbols.

Lemma 8.3.8 *A language L is generated by a context-sensitive grammar iff it is accepted by a linear-bounded automaton.*

The class of context-sensitive languages is very large. The main problem is that no practical methods for constructing parsers from csg's are known.