Chap. 5 Context-Free Grammars

5.1 Context-Free Grammars

5.1.1 An Informal Example

madamimadam "*Madam, I'm Adam*"

A string w is ***palindrome****, if and only if* $w = w^R$.

Palindromes over {0, 1}

- ***basis***: ε, *0, and 1 are palindromes.*
- ***induction***: *If w is a palindrome, so are 0w0 and 1w1.*
  No other string is palindrome.

Context-Free Grammar

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

5.1.2 Definition of Context-Free Grammars

*A guadruple $G = (N, \Sigma, P, S)$ is a **context-free grammar**, if*

1. *N is a **finite** set of **nonterminals**(**variables**, **syntactic categories**),*
2. *Σ is a **finite** set of **terminal** symbols, where $N \cap \Sigma = \varnothing$,*
3. *P is a **finite** set of **productions**(**rules**),*
   where each production is a pair (A, α), written $A \to \alpha$,
   $A \in N$ — ***left part**(**head**) of production*
   $\alpha \in (N \cup \Sigma)^*$, — ***right part**(**body**) of production*
4. *$S \in N$ is a distinguished variable, called **start**(**axiom**) **symbol**.*

Example 5.2

$$G_{pal} = (\{P\}, \{0, 1\}, \{P \to \varepsilon, P \to 0, P \to 1, P \to 0P0, P \to 1P1\}, P)$$

*We write $A \to \alpha_1 \mid \ldots \mid \alpha_n \in P$ **instead** of $A \to \alpha_1, \ldots, A \to \alpha_n \in P$.*

Example 5.2'

$$G_{pal} = (\{P\}, \{0, 1\}, \{P \to \varepsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1\}, P)$$

Example 5.3 regular expressions *over* {**a**, **b**, **0**, **1**}

$E \to E + E \mid EE \mid E^{*} \mid (E) \mid B$ ***induction***

$B \to \underline{\varepsilon} \mid \varnothing \mid \mathbf{a} \mid \mathbf{b} \mid \mathbf{0} \mid \mathbf{1}$ ***basis***

Note that $\underline{\varepsilon}$ *is not the* ***empty string*** *but a* ***symbol*** *for regular expression.*

$N = \{E, B\}$

$\Sigma = \{\underline{\varepsilon}, \varnothing, \mathbf{a}, \mathbf{b}, \mathbf{0}, \mathbf{1}, +, {}^{*}, (,)\}$

Example 5.3 regular expression *revisited*

$E \to E + E \mid EE \mid E^{*} \mid (E) \mid \underline{\varepsilon} \mid \varnothing \mid \mathbf{a} \mid \mathbf{b} \mid \mathbf{0} \mid \mathbf{1}$

5.1.3 Derivation Using a Grammar

Let $\alpha, \gamma \in (N \cup \Sigma)^*$ *and* $B \in N$, *and* $B \to \beta \in P$ *be a* ***production***.

We say string $\alpha B \gamma$ ***directly derives*** $\alpha\beta\gamma$ *in CFG G, written*

$\alpha B \gamma \Rightarrow_G \alpha\beta\gamma$, *we may omit G when it is understood,* $\alpha B\gamma \Rightarrow \alpha\beta\gamma$.

$\Rightarrow \subseteq (N \cup \Sigma)^* \times (N \cup \Sigma)^*$ *a* ***binary relation on*** $(N \cup \Sigma)^*$.

$\to \subseteq \Rightarrow$ $\Rightarrow$ *is an* ***induced*** *binary relation from* $\to$.

Note that $\to$ *is* ***finite*** *but* $\Rightarrow$ *is* ***infinite***.

$\to$ *is an* ***extension*** *of* $\Rightarrow$.

Recursive *definition of* $\Rightarrow^i$.

1. $\alpha \Rightarrow^0 \alpha, \forall \alpha \in (N \cup \Sigma)^*$. ***basis***
2. *For* $n \geq 1$, *if* $\alpha \Rightarrow^n \beta$, *and* $\beta \Rightarrow \gamma$, *then* $\alpha \Rightarrow^{n-1} \gamma$. ***recursion***

Definition of $\Rightarrow^*$.

$\Rightarrow^* = \cup_{i \in N_0} \Rightarrow^i$. ***reflexive transitive closure*** *of* $\Rightarrow$.

We say α ***derives*** β*, if* $\alpha \Rightarrow^* \beta$ *for some* $\alpha, \beta \in (N \cup \Sigma)^*$.
We say α *is a* **sentential form** *of G, if* $S \Rightarrow^* \alpha$ *for some* $\alpha \in (N \cup \Sigma)^*$.
We say w is a **sentence** *of G, if* $S \Rightarrow^* w$ *for some* $w \in \Sigma^*$.

The ***language*** *of G, denoted L(G), is* $L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$.

A language L is ***context-free****, if there is a cfg G such that L = L(G).*

Notational conventions for CFG

$a, b, c, \ldots \in \Sigma$	***terminal symbols***
$A, B, C, \ldots \in N$	***variable symbols***
$X, Y, Z, \ldots \in N \cup \Sigma$	***general symbols***
$x, y, z, \ldots \in \Sigma^*$	***terminal strings***
$\alpha, \beta, \gamma, \ldots \in (N \cup \Sigma)^*$	***general strings***

Derivation *of CFG is* ***nondeterministic***

1. *Which variable to be replaced*
2. *Which right hand side of the rule to be replaced*

leftmost derivation, $\Rightarrow_{lm}$, *to replace* ***leftmost*** *variable*

$$S \Rightarrow_{lm}^{*} xB\gamma \Rightarrow_{lm} x\beta\gamma \Rightarrow_{lm}^{*} xy\gamma \Rightarrow_{lm}^{*} xyz$$

where $x, y, z \in \Sigma^{*}$, $\gamma \in (N \cup \Sigma)^{*}$, $A \rightarrow \beta \in P$.

rightmost derivation, $\Rightarrow_{rm}$, *to replace* ***rightmost*** *variable*

$$S \Rightarrow_{rm}^{*} \alpha Bz \Rightarrow_{rm} \alpha\beta z \Rightarrow_{rm}^{*} \alpha yz \Rightarrow_{rm}^{*} xyz$$

where $x, y, z \in \Sigma^{*}$, $\alpha \in (N \cup \Sigma)^{*}$, $A \rightarrow \beta \in P$.

Note that $\Rightarrow_{lm}, \Rightarrow_{rm} \subseteq \Rightarrow$.

5.2 Parse Trees

Let G = (N, Σ, P, S) be a cfg. The ***parse tree*** *for G are trees*

1. *Each* ***interior*** *node is labelled by a* ***variable*** $A \in N$
2. *Each* ***leaf*** *node is labelled by a* ***terminal*** $a \in \Sigma$ *or* ε.
3. *If an interior node is labelled A and its children are labelled*

    $X_1, X_2, \ldots, X_n$ *from left to right*

    $A \rightarrow X_1X_2\ldots X_n \in P$.

yield *of a tree*

concatenation *of* ***leaves*** *of a tree* ***from left to right***

Recursive definition*(****Top Down construction****) of parse tree.*

Basis *({S}, ∅, **S**) is a parse tree.*

Recursion *Let (V, E, **S**) be parse trees. If* $A \in V$ *is a node and* $(A \in N)$

$A \rightarrow X_1X_2\ldots X_n \in P$. *Then (V', E', **S**) is a* ***new*** *parse tree with root **S** where*

$V' = V \cup \{X_1, X_2, \ldots, X_n\}$ *and* $E' = E \cup \{(A, X_i) \mid 1 \le \forall i \le n\}$.

*Two **futures** of the **new** leaf nodes X_i's in the parse tree (V', E', **S**).*

*i) $X_i \in \Sigma \rightarrow$ the node X_i remains as a **leaf** node.*

*ii) $X_i \in N \rightarrow$ the node X_i will be an **interior** node(a **root** of **subtree**).*

***Recursive definition2(Bottum Up construction)** of parse tree.*

Basis $\forall X \in N \cup \Sigma$: $(\{X\}, \emptyset, X)$ *can be(?) parse trees.*

***Recursion** Let* $A \rightarrow X_1X_2\ldots X_n \in P$ *where* $1 \le \forall i \le n$: $X_i \in N \cup \Sigma$ *and*

$(V_1, E_1, X_1), (V_2, E_2, X_2), \ldots, (V_n, E_n, X_n)$ *be **new** parse trees.*

*Then (V, E, A) is a **new** parse tree where*

$V = \{A\} \cup \cup_{i\in\{1, 2, \ldots, n\}} V_i = \{A\} \cup V_1 \cup V_2 \cup \ldots \cup V_n$ *and*

$E = \cup_{i\in\{1, 2, \ldots, n\}} \{(A, X_i)\} \cup E_1 \cup E_2 \cup \ldots \cup E_n.$

*See details for **right parser** in the supplement 2 TP*

Following four statements are ***equivalent*** *for some* ***terminal*** *string* $x \in \Sigma^*$.

(1) $A \Rightarrow^* x$,

(2) $A \Rightarrow_{lm}^* x$,

(3) $A \Rightarrow_{rm}^* x$,

(4) *There is a parse tree with* ***root*** *A and* ***yield*** *x.*

Proof (2) $\Rightarrow$ (1), (3) $\Rightarrow$ (1) are trivial ($\Rightarrow_{lm}, \Rightarrow_{rm} \subseteq \Rightarrow$).

(1) $\Rightarrow$ (4): Thm. 5.12

(4) $\Rightarrow$ (2), (4) $\Rightarrow$ (3): Thm 5.14 and 5.16

Theorem 5.12 *If $A \Rightarrow^* x$, then there is a parse tree with* ***root*** *A and* ***yield*** *x*

Proof *Induction on* ***number*** *of* ***derivations***

Basis $A \Rightarrow^1 x$, $A \rightarrow x \in P$. $\therefore$ *Parse tree in Fig. 5.8(p187).*

Induction *Assume* $A \rightarrow X_1X_2\ldots X_n \in P$, $1 \le^{\forall} i \le n$: $X_i \Rightarrow^{k_i} x_i$, *and* $x = x_1x_2\ldots x_n$

1) *If* $X_i \in \Sigma$, $X_i = x_i$, $X_i \Rightarrow^{k_i} (\Rightarrow^0; =)\ x_i$. $\therefore$ ***parse tree*** *with* ***leaf*** $x_i \in \Sigma$.

2) *If* $X_i \in N$, $X_i \Rightarrow^{k_i} x_i$. $\therefore$ ***parse tree*** *with* ***root*** X_i *and* ***yield*** x_i*.(IH)*

$\therefore A \Rightarrow X_1X_2\ldots X_n \Rightarrow^{k_1} x_1X_2\ldots X_n \Rightarrow^{k_2} x_1x_2\ldots X_n \Rightarrow^{k_3} \ldots \Rightarrow^{k_n} x_1x_2\ldots x_n$.

$\therefore$ *If* $A \Rightarrow^m x_1x_2\ldots x_n = x (m \ge 1)$ *where* $\Sigma_n k_i = m-1$, *then*

parse tree *with* ***root*** *A and* ***yield*** *x.(Fig. 5.9; p188)*

Theorem 5.14 *If there is a parse tree with* ***root*** *A and* ***yield*** *x,* $A \Rightarrow_{lm}^{*} x$.

Proof *Induction on* ***height*** *of a tree*

Basis *Parse tree with height 1, in Fig. 5.8.* $A \rightarrow x \in P$. $A \Rightarrow_{lm} x$.

Induction *Consider a* ***parse tree*** *with* ***root*** *A, height m, and sons* $X_1, X_2, \ldots, X_n$ *from left to right, and* ***yield*** $x = x_1x_2\ldots x_n$.*(Fig. 5.9)*

1) If $X_i \in \Sigma$, $X_i = x_i$. $X_i \Rightarrow_{lm}^{0} x_i$.

2) If $X_i \in N$, $X_i \Rightarrow_{lm}^{+} x_i$.*(IH; with height is less than m)*

$$A \Rightarrow_{lm} X_1X_2\ldots X_n \Rightarrow_{lm}^{*} x_1X_2\ldots X_n \Rightarrow_{lm}^{*} x_1x_2\ldots X_n \Rightarrow_{lm}^{*} x_1x_2\ldots x_n.$$

Theorem 5.16 *If there is a parse tree with* ***root*** *A and* ***yield*** *x,* $A \Rightarrow_{rm}^{*} x$.

Proof *Induction on* ***height*** *of a tree(similar to leftmost derivation)*

$$A \Rightarrow_{rm} X_1X_2\ldots X_n \Rightarrow_{rm}^{*} X_1X_2\ldots x_n \Rightarrow_{rm}^{*} X_1x_2\ldots x_n \Rightarrow_{rm}^{*} x_1x_2\ldots x_n.$$

5.4 Ambiguity in Grammars and Languages

G_1: $E \rightarrow E + E \mid E * E \mid a \mid (E)$

$E \Rightarrow_{lm} E + E \Rightarrow_{lm} a + E \Rightarrow_{lm} a + E * E \Rightarrow_{lm} a + a * E \Rightarrow_{lm} a + a * a.$

$E \Rightarrow_{lm} E * E \Rightarrow_{lm} E + E * E \Rightarrow_{lm} a + E * E \Rightarrow_{lm} a * E \Rightarrow_{lm} a + a * a.$

A grammar G is said to be ***ambiguous****, if* $^{\exists}x \in L(G)$ *.э.*

x has more than one ***parse trees****(****syntactic structure****),*

(x has more that one ***leftmost****(****rightmost****) derivation sequences)*

otherwise, ***unambiguous****.*

Derivation revisited

We may write $\alpha A\beta \Rightarrow^{r} \alpha\gamma\beta$, *if* $r = A \rightarrow \gamma \in P$.

Recursive extension of derivation with rules(rule string)

$\alpha \Rightarrow^{\varepsilon} \alpha$, *for* $\alpha \in (N \cup \Sigma)^{*}$, *and*

$\alpha \Rightarrow^{\pi r} \gamma$, *if* $\alpha \Rightarrow^{\pi} \beta$, $\beta \Rightarrow^{r} \gamma$ *for* $\pi \in P^{*}$, $r \in P$, $\alpha,\beta,\gamma \in (N \cup \Sigma)^{*}$.

Parser *of a grammar G,*

$\forall x \in \Sigma^*$, *if* $x \in L(G)$ ***syntactic structure****(parse tree),*

otherwise say NO.

Is the parser for G is ***deterministic****?*

Not always!

It is ***undecidable*** *whether G is* ***ambiguous*** *or not.*

If G is ***unambiguous****, the parser for G may be* ***deterministic or not****.*
If G is ***ambiguous****, the parser for G is* ***nondeterministic****.*
If the parser for G is ***deterministic****, G is* ***unambiguous****.*

	parser	***structure***
regular	***deterministic***	***linear***
context-free	***nondeterministic***	***hierarchical***

Deterministic parsing *of* ***context-free languages***

If $S \Rightarrow_{lm}^{\pi} x$ *for* $x \in \Sigma^*$, $\pi \in P^*$ *is called the* ***left parse*** *of the sentence x.*

If $S \Rightarrow_{rm}^{\pi} x$ *for* $x \in \Sigma^*$, $\pi^R \in P^*$ *is called the* ***right parse*** *of the sentence x.*

parse tree ⇔ ***left parse*** ⇔ ***right parse*** ⇔ ***syntactic structure***

left(right) parser*:* ***left(right)*** *parse*

LL(k): ***Left-to-right scan*** *in* ***Leftmost derivation*** *with k-lookahead symbols*
LR(k): ***Left-to-right scan*** *in* ***Rightmost derivation*** *with k-lookahead symbol*

Left parser *same direction in scan and derivation*
normal order, top-down parsing*(LL parsing)*
Right parser *different direction in scan and derivation*
reversed order, bottom-up parsing*(LR, LALR, SLR parsing)*

Removing ambiguity in the grammar

*Assume that precedence of * is higher than that of +, and*
*+ and * are left associative.*

G_2: $E \to E + T \mid T * F \mid a \mid (E)$
$T \to T * F \mid a \mid (E)$
$F \to a \mid (E)$

G_3: $E \to E + T \mid T$
$T \to T * F \mid F$
$F \to a \mid (E)$

$|G| = \Sigma_{A\to\alpha\in P} |A| + |\alpha| = \Sigma_{A\to\alpha\in P} |\alpha| + 1$ **Size** *of a grammar*

$|G_2| = 14+10+6 = 30$, $|G_3| = 6+6+6 = 18$, $|G_1| = 4+4+2+4 = 14$.

$A \to B$ is called ***unit production****, if $A, B \in N$.*

*A **context free language** L is said to be **inherently ambiguous**,*
*if **every** cfg G .϶. L = L(G) is **ambiguous**.*

$L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$ *is **inherently ambiguous**.*

Consider G:

$$S \rightarrow AB \mid C$$
$$A \rightarrow aAb \mid ab$$
$$B \rightarrow cBd \mid cd$$
$$C \rightarrow aCd \mid aDd$$
$$D \rightarrow bDc \mid bc$$

and the sentence $a^n b^n c^n d^n$.

$$S \Rightarrow_{lm} AB \Rightarrow_{lm} aAbB \Rightarrow_{lm}^{*} a^{n-1}Ab^{n-1}B \Rightarrow_{lm} a^n b^n B$$
$$\Rightarrow_{lm} a^n b^n cBd \Rightarrow_{lm}^{*} a^n b^n c^{n-1}Bd^{n-1} \Rightarrow_{lm} a^n b^n c^n d^n.$$
$$S \Rightarrow_{lm} C \Rightarrow_{lm} aCd \Rightarrow_{lm}^{*} a^{n-1}Cd^{n-1} \Rightarrow_{lm} a^n D d^n$$
$$\Rightarrow_{lm} a^n bDcd^n \Rightarrow_{lm}^{*} a^n b^{n-1}Dc^{n-1}d^n \Rightarrow_{lm} a^n b^n c^n d^n.$$