Formal Languages Part 2
Context Free Grammars

Context-Free Grammars

- Example: an English sentence:
  
  
  Sentence:
  Old        Harry    jogs

  Constituents:       subject           predicate
  Word Class:   adjective   noun    verb

  <sentence>
  <predicate>
  <subject>
  <verb>

- A grammar is used to describe the syntax.

  BNF (Backus-Naur form) 1960 (meta-language to describe languages):
  
  <sentence> → <subject><predicate>
  <subject> → <adjective><noun>
  <predicate> → <verb>
  <adjective> → old | big | strong | ...
  <noun> → Harry | brother | ...
  <verb> → jogs | snores | sleeps | ...

Gratmars, cont.

- <sentence> is a start symbol.
- Symbols to the left of "→" are called nonterminals.
- Symbols not surrounded by "< >" are terminals.
- Each line is a production.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; ... &gt;</td>
<td>syntactic classes</td>
</tr>
<tr>
<td>→</td>
<td>&quot;consists of&quot;, &quot;is&quot; (also &quot;::=&quot;)</td>
</tr>
<tr>
<td></td>
<td>&quot;or&quot;</td>
</tr>
</tbody>
</table>

A Grammar can be used to Produce or Derive Sentences

- Example: <sentence> ⇒* Old Harry jogs
  where <sentence> is the start symbol and ⇒* means derivation in zero or more steps.

Example Derivation:

  <sentence> ⇒ <subject> <predicate>
  ⇒ <adjective> <noun> <predicate>
  ⇒ Old <noun> <predicate>
  ⇒ Old Harry <predicate>
  ⇒ Old Harry <verb>
  ⇒ Old Harry jogs

Definition: CFG (Context-free grammar)

- A CFG (Context-free grammar) is a quadruple (4 parts):

  G = < N, Σ, P, S >

  where
  N : Nonterminals.
  Σ : terminal Symbols.
  P : rules, Productions of the form
  A → a where A ∈ N and
  a ∈ (N ∪ Σ)*
  S the Start symbol, a nonterminal, S ∈ N.
- (Sometimes V = N ∪ Σ is used, called the vocabulary.)

Notational Conventions

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>α, β, γ ∈ V*</td>
<td>string of terminals and nonterminals</td>
</tr>
<tr>
<td>A, B, C ∈ N</td>
<td>nonterminals</td>
</tr>
<tr>
<td>a, b, c ∈ Σ</td>
<td>terminal symbols</td>
</tr>
<tr>
<td>u, v, w, x, y, z ∈ Σ*</td>
<td>string of terminals</td>
</tr>
</tbody>
</table>
Derivations

- Derivation
  - Derivation $\alpha \Rightarrow \beta$ (pronounced "$\alpha$ derives $\beta$")
  - Formally: $\gamma A \delta \Rightarrow \gamma \delta \theta$ if we have $A \Rightarrow \delta$
  - Example: $<\text{number}> \Rightarrow_{lm} <\text{no}> \Rightarrow_{lm} <\text{no}> <\text{digit}> \Rightarrow_{lm} <\text{no}> 2 \Rightarrow_{lm} <\text{digit}> 2 \Rightarrow_{lm} 12$
  - $<\text{no}> \Rightarrow <\text{no}>$ direct derivation.
  - $<\text{no}> \Rightarrow_+ 12$ several derivations (zero or more).
  - $<\text{no}> \Rightarrow_+ 12$ several derivations (one or more).

Given $G = \langle N, \Sigma, P, S \rangle$ the language generated by $G$ can be defined as $L(G)$:

$L(G) = \{ w \mid S \Rightarrow^* w \text{ and } w \in \Sigma^* \}$

Sentential form, Sentence

- Sentential form
  - A string $\alpha$ is a sentential form in $G$ if $S \Rightarrow^* \alpha$ and $\alpha \in \Sigma^*$ (string of terminals and/or nonterminals)
  - Example: $<\text{no}> <\text{digit}>$ is a sentential form in $G(<\text{number}>)$.

- Sentence
  - $w$ is a sentence in $G$ if $S \Rightarrow^+ w$ and $w \in \Sigma^*$.
  - Example: $12$ is a sentence in $G(<\text{number}>)$.

Left and Right Derivations

- Left derivation
  - $\Rightarrow_{lm}$ means that we replace the leftmost nonterminal by some appropriate right side.
  - Left sentential form
    - A sentential form which is part of a leftmost derivation.

Right derivation (canonical derivation)

- $\Rightarrow_{rm}$ means that we replace the rightmost nonterminal by some appropriate right side.
  - Right sentential form
    - A sentential form which is part of a rightmost derivation.

Rightmost Derivation, Handle

- Reverse rightmost derivation
  - $12 \Rightarrow_{rm} <\text{digit}> 2 \Rightarrow_{rm} <\text{no}> <\text{digit}> <\text{no}> \Rightarrow_{rm} <\text{no}> <\text{digit}> <\text{no}> \Rightarrow_{rm} <\text{no}> 2 \Rightarrow_{rm} <\text{digit}> 2 \Rightarrow_{rm} 12$

- Handles
  - Consist of two parts:
    - 1. A production $A \Rightarrow \beta$
    - 2. A position $A \Rightarrow \beta$
    - 3. A production $A \Rightarrow \beta$
    - 4. A position $A \Rightarrow \beta$

- Example: The handle of $<\text{no}> 2$ is the production $<\text{digit}> \Rightarrow 2$ and the position after $<\text{no}>$ because:
  - $<\text{no}> \Rightarrow_{lm} \alpha A \Rightarrow_{lm} \alpha \beta w$, the production $A \Rightarrow \beta$ together with the position after $\alpha$ is a handle of $\beta w$.

- Informally: a handle is what we reduce to what and where to get the previous sentential form in a rightmost derivation.

Reduction

- Reduction of a grammar rule

  In reverse right derivation, find a right side in some rule according to the grammar in the given right sentential form and replace it with the corresponding left side, i.e., nonterminal.

Parse trees (derivation trees)

- A parse tree can correspond to several different derivations.

Example Grammar:

```
<\text{number}>
  /   \
<no>   <\text{digit}>

<\text{digit}>
  /   \
0|1|2|3|4|5|6|7|8|9
```

Parse tree for 12
Ambiguous Grammars

- A grammar G is ambiguous if a sentence in G has several different parse trees.

\[
\begin{align*}
E & \rightarrow E + E \\
& \quad | E \ast E \\
& \quad | id
\end{align*}
\]

- id&id has two different parse trees.

\[
\begin{align*}
E & \rightarrow \quad \text{id/id/id}
\end{align*}
\]

Rewriting to Unambiguous Grammar

- Rewrite the grammar to make it unambiguous:
  - +, * are to have the right priority and
  - +, * are to be left associative while
  - ↑ is to be right associative.

\[
\begin{align*}
E & \rightarrow E + T \\
& \quad | T \\
T & \rightarrow T \ast F \\
& \quad | F \\
F & \rightarrow P \uparrow F \\
P & \rightarrow id
\end{align*}
\]

Example Palindrome Grammars

- Palindrome: a string that is symmetrical around its center

\[
\begin{align*}
S & \rightarrow \text{a S b} \\
& \quad | a \ b
\end{align*}
\]

Another example: Grammar describing binary palindromes of odd lengths >= 1:

\[
\begin{align*}
S & \rightarrow 0 S 0 \\
& \quad | 1 S 1 \\
& \quad | 0 \\
& \quad | 1
\end{align*}
\]

Example derived strings: 0, 1, 000, 010, 111

Small Parse Tree Exercise

Example Parse Tree:

- Example: The following grammar generates \{ anbn | n >= 1 \}.

\[
\begin{align*}
S & \rightarrow \text{a S b} \\
& \quad | a \ b
\end{align*}
\]

Binary Palindrome Exercise

- Example Excerpt from a Pascal Grammar

\[
\begin{align*}
\text{goal} & \rightarrow \text{<progdecl> ; <block>} \\
\text{<progdecl>} & \rightarrow \text{<prog_hedr> ; <block>} \\
\text{<prog_hedr>} & \rightarrow \text{program <idname> ( <idname_list> )} \\
& \quad | \text{program <idname>} \\
\text{<block>} & \rightarrow \text{<decls> begin <stat_list> end} \\
\text{<label>} & \rightarrow \text{<label decl> ; <label decl>} \\
\text{<label decl>} & \rightarrow \text{<labelid> = <const> ; <label decl>} \\
\text{<labelid>} & \rightarrow \text{<id> ; <id> = <const> ; <id> = <type> ; <id> = <type> ; <id> = <type>}
\end{align*}
\]
Example

Excerpt from a Pascal Grammar, Cont.

\[
\begin{align*}
\langle \text{proc} \rangle & \rightarrow \text{procedure} \langle \text{phead_c} \rangle \text{ forward } \; | \; \langle \text{phead_c} \rangle \langle \text{block} \rangle \\
\langle \text{phead} \rangle & \rightarrow \langle \text{phead_c} \rangle \; | \; \langle \text{phead_c} \rangle \langle \text{block} \rangle \\
\langle \text{phead_c} \rangle & \rightarrow \langle \text{phead} \rangle \\
\langle \text{fhead} \rangle & \rightarrow \langle \text{idname} \rangle \langle \text{params} \rangle : \langle \text{type_id} \rangle \\
\langle \text{params} \rangle & \rightarrow ( \langle \text{param_list} \rangle ) \\
\langle \text{param} \rangle & \rightarrow \text{var} \langle \text{par_decl} \rangle \\
\langle \text{par_decl} \rangle & \rightarrow \langle \text{id_list} \rangle : \langle \text{type_id} \rangle \\
\langle \text{param_list} \rangle & \rightarrow ( \langle \text{param} \rangle ; \langle \text{param} \rangle ) \\
\langle \text{id_list} \rangle & \rightarrow \langle \text{id} \rangle \\
\end{align*}
\]