Grammar: Features and Unification
Plan for the Talk

- Problems with CFG (PCFG)
  - Features Structure
    - Attribute-value Matrix (AVM)
  - Unification
  - Grammar formalisms based on unification
Agreement

- Constraints that hold among various constituents.
- For example, in English, determiners and the head nouns in NPs have to agree in their number.

Which of the following cannot be parsed by the rule

\[ \text{NP} \rightarrow \text{Det Nominal} \]?

- (O) This flight
- (O) Those flights
- (X) This flights
- (X) Those flight
Agreement

- Constraints that hold among various constituents.
- For example, in English, determiners and the head nouns in NPs have to agree in their number.

- Which of the following cannot be parsed by the rule

\[
\text{NP} \rightarrow \text{Det Nominal}
\]

\[\rightarrow\] This rule does not handle agreement! (The rule does not detect whether the agreement is correct or not.)

- (O) This flight
- (O) Those flights
- (X) This flights
- (X) Those flight
Problem with CFG/PCFG

- Our earlier NP rules are clearly deficient since they don’t capture the agreement constraint
  - \( NP \rightarrow Det \ Nominal \)
    - Accepts, and assigns correct structures, to grammatical examples (this flight)
    - But it's also happy with incorrect examples (*these flight)
  - Such a rule is said to overgenerate.
  - We’ll come back to this in a bit
Verb Phrases

- English VPs consist of a head verb along with 0 or more following constituents which we’ll call *arguments*.

\[
\begin{align*}
VP & \rightarrow \text{Verb} \quad \text{disappear} \\
VP & \rightarrow \text{Verb NP} \quad \text{prefer a morning flight} \\
VP & \rightarrow \text{Verb NP PP} \quad \text{leave Boston in the morning} \\
VP & \rightarrow \text{Verb PP} \quad \text{leaving on Thursday}
\end{align*}
\]
Subcategorization

- *John sneezed the book
- *I prefer United has a flight
- *Give with a flight

As with agreement phenomena, we need a way to formally express the constraints!
Subcategorization

- **Sneeze**: John sneezed
- **Find**: Please find [a flight to NY]_{NP}
- **Give**: Give [me]_{NP}[a cheaper fare]_{NP}
- **Help**: Can you help [me]_{NP}[with a flight]_{PP}
- **Prefer**: I prefer [to leave earlier]_{TO-VP}
- **Told**: I was told [United has a flight]_{S}
- ...
Subcategorization

- But, even though there are many valid VP rules in English, not all verbs are allowed to participate in all those VP rules.
- We can subcategorize the verbs in a language according to the sets of VP rules that they participate in.
- This is a modern take on the traditional notion of transitive/intransitive.
- Modern grammars may have 100s or such classes.
Problem with CFG/PCFG

- Right now, the various rules for VPs *overgenerate*.
  - They permit the presence of strings containing verbs and arguments that don’t go together
  - For example
    - VP -> V NP therefore
      - *Sneezed the book* is a VP since “sneeze” is a verb and “the book” is a valid NP
Possible CFG Solution

- Possible solution for agreement.
- Can use the same trick for all the verb/VP classes.

- SgS -> SgNP SgVP
- PlS -> PlNP PlVP
- SgNP -> SgDet SgNom
- PlNP -> PlDet PlNom
- PlVP -> PIV NP
- SgVP -> SgV Np
- ...

...
CFG Solution for Agreement

- **Pro:**
  - It works and stays within the power of CFGs

- **Con:**
  - Loss of generalization – “apple” and “apples” are treated as if they are two separate words
  - And it doesn’t scale all that well because of the interaction among the various constraints explodes the number of rules in our grammar.
Non-CFG Solution for Agreement

- Add “constraints” to each rule
  - S -> NP VP
    constraint: only if the number of NP is equal to the number of the VP

- Instead of replicating rules...
  - SgS -> SgNP SgVP
  - PIS -> PINp PIVP
  - SgNP -> SgDet SgNom
  - PINP -> PIDet PINom
  - PIVP -> PIV NP
  - SgVP -> SgV Np
  - ...

Plan for the Talk

- Problems with CFG (PCFG)
- Features Structure
  - Attribute-value Matrix (AVM)
- Unification
- Grammar formalisms based on unification
Feature Structure

- “Features” in formal grammar

- “Features” in machine learning

- Attribute-value Matrix (AVM)
  - Feature Path
  - Reentrant structure
Feature Structure

This feature structure is used in many grammar formalism that goes beyond CFG, such as

- Lexical Functional Grammar (LFG) (Bresnan, 1982)
- Construction Grammar (Kay and Fillmore, 1999)
- Unification Categorial Grammar (Uszkoreit, 1986)
Attribute-value matrix (AVM)

Definition:

\[
\begin{pmatrix}
\text{FEATURE}_1 & \text{value}_1 \\
\text{FEATURE}_2 & \text{value}_2 \\
\text{....} & \text{....} \\
\text{FEATURE}_n & \text{value}_n
\end{pmatrix}
\]

For example:

\[
\begin{pmatrix}
\text{NUMBER} & \text{sg}
\end{pmatrix}
\]
Attribute-value matrix (AVM)

More Examples:

\[
\begin{array}{c|c}
\text{CAT} & \text{NP} \\
\text{NUMBER} & \text{sg} \\
\text{PERSON} & \text{3rd}
\end{array}
\]
Attribute-value matrix (AVM)

Hierarchical Structure: “value” can be another AVM object

```
CAT
NUMBER
PERSON
3rd
```

```
CAT
AGREEMENT
NP
NUMBER
PERSON
sg
3rd
```
Feature Path

Feature Path: a sequence of features in the feature structure (AVM) leading to a particular value
Feature Path

Feature Path: a sequence of features in the feature structure (AVM) leading to a particular value
Attribute-value matrix (AVM)

Reentrant Structure:

```
[CAT]
[HEAD]
[S]

AGREEMENT [1]

NUMBER

PERSON 3rd

SUBJECT

AGREEMENT [1]

sg
```
Reentrant Structure:

CAT
HEAD

( S
AGREEMENT [1]

SUBJECT

NUMBER
sg

PERSON
3rd

AGREEMENT [1]

Feature Path:
Feature Structure

- “Features” in formal grammar
- “Features” in machine learning

- Attribute-value Matrix (AVM)
  - Feature Path
  - Reentrant structure

- This feature structure is used in many grammar formalism that goes beyond CFG, such as HPSG, LFG
Plan for the Talk

• Problems with CFG (PCFG)
• Features Structure
  • Attribute-value Matrix (AVM)
• Unification
• Grammar formalisms based on unification
Unification of Feature Structure

- Unification of two feature structure (AVM) finds the most general feature structure that is compatible with the two given AVMs.

- \([ \text{NUMBER sg} ] \cup [ \text{NUMBER sg} ] =\]

- \([ \text{NUMBER sg} ] \cup [ \text{NUMBER pl} ] =\]

- \([ \text{NUMBER sg} ] \cup [ \text{NUMBER [ ] } ] =\]
Unification of Feature Structure

- Unification of two feature structure (AVM) finds the most general feature structure that is compatible with the two given AVMs.

- \([\text{NUMBER sg }] \cup [\text{NUMBER sg}] = [\text{NUMBER sg}]\)

- \([\text{NUMBER sg}] \cup [\text{NUMBER pl}] \rightarrow \text{Fails!}\)

- \([\text{NUMBER sg}] \cup [\text{NUMBER [ ]}] = [\text{NUMBER sg}]\)
Unification of Feature Structure

- Unification of two feature structure (AVM) finds the most general feature structure that is compatible with the two given AVMs.

- \([ \text{NUMBER sg} ] \cup [ \text{PERSON 3rd} ] = \)
Unification of Feature Structure

- Unification of two feature structure (AVM) finds the most general feature structure that is compatible with the two given AVMs.

- $[ \text{NUMBER } \text{sg} ] \cup [ \text{PERSON } 3\text{rd} ] = \begin{cases} \text{NUMBER } \text{sg} \\ \text{PERSON } 3^{\text{rd}} \\ \text{CATEGORY NP} \end{cases}$
Unification of Feature Structure

- Unification of two feature structure (AVM) finds the most general feature structure that is compatible with the two given AVMs.

- \([ \text{NUMBER sg} ] \cup [ \text{PERSON 3rd} ] = \begin{cases} \text{NUMBER sg} \\ \text{PERSON 3rd} \\ \text{CATEGORY NP} \end{cases} \)?
Unification of Feature Structure

- Unification of two feature structure (AVM) finds the **most general** feature structure that is compatible with the two given AVMs.

  \[
  [ \text{NUMBER sg} ] \cup [ \text{PERSON 3rd} ] = \begin{cases} 
  \text{NUMBER sg} \\
  \text{PERSON 3rd}
  \end{cases}
  \]
Unification of Feature Structure

\[
\begin{align*}
\text{SUBJECT} & \quad \left\langle \begin{array}{c}
\text{AGREEMENT [1]} \\
\text{NUMBER} \quad \text{sg} \\
\text{PERSON} \quad 3^{\text{rd}}
\end{array} \rightangle \\
\text{U} & \quad \left\langle \begin{array}{c}
\text{SUBJECT} \\
\text{AGREEMENT} \\
\text{PERSON} \quad 3^{\text{rd}}
\end{array} \rightangle \\
= & \quad \left\langle \begin{array}{c}
\text{NUMBER} \quad \text{sg} \\
\text{PERSON} \quad 3^{\text{rd}}
\end{array} \rightangle
\end{align*}
\]
Unification of Feature Structure

\[
\begin{align*}
\text{SUBJECT} & \quad \left\{ \begin{array}{c}
\text{AGREEMENT} \ [1] \\
\text{NUMBER} & \text{sg} \\
\text{PERSON} & \text{3}^{rd}
\end{array} \right. \\
U & \quad \left\{ \begin{array}{c}
\text{SUBJECT} \\
\text{AGREEMENT} \ [1]
\end{array} \right. \\
= & \quad \left\{ \begin{array}{c}
\text{AGREEMENT} \ [1] \\
\text{NUMBER} & \text{sg} \\
\text{PERSON} & \text{3}^{rd}
\end{array} \right.
\end{align*}
\]
Unification of Feature Structure

\[
\begin{align*}
\text{AGREEMENT [1]} \\
\text{SUBJECT} & \quad \left( \text{AGREEMENT [1]} \right) \\
U \quad \text{SUBJECT} & \quad \left( \text{AGREEMENT} \left( \text{PERSON} \left( 3^{\text{rd}} \right) \right) \right) \\
& = 
\end{align*}
\]
Unification of Feature Structure

\[
\text{AGREEMENT} [1] \\
\text{SUBJECT} \quad \text{AGREEMENT} [1] \\
U \quad \text{SUBJECT} \quad \text{AGREEMENT} \quad \text{PERSON} \; 3^{\text{rd}} \\
\quad \quad \text{NUMBER} \; \text{sg} \\
= \text{AGREEMENT} [1] \\
\text{SUBJECT} \quad \text{AGREEMENT} [1] \quad \text{PERSON} \; 3^{\text{rd}} \\
\quad \quad \text{NUMBER} \; \text{sg}
\]
Unification of Feature Structure

\[
\begin{align*}
\text{U} & \left(\begin{array}{c}
\text{AGREEMENT} \ [1] \\
\text{NUMBER} & \text{sg} \\
\text{PERSON} & 3^{\text{rd}} \\
\text{SUBJECT} \\
\text{AGREEMENT} \ [1] \\
\end{array}\right) \\
= & \left(\begin{array}{c}
\text{AGREEMENT} \\
\text{NUMBER} & \text{sg} \\
\text{PERSON} & 3^{\text{rd}} \\
\text{SUBJECT} \\
\text{AGREEMENT} \\
\text{PERSON} & 3^{\text{rd}} \\
\text{NUMBER} & \text{pl} \\
\end{array}\right)
\end{align*}
\]
Unification of Feature Structure

\[
\begin{align*}
&\text{SUBJECT} = \left( \begin{array}{c}
\text{AGREEMENT} \ [1] \\
\text{NUMBER} \ sg \\
\text{PERSON} \ 3^{rd}
\end{array} \right) \\
&U = \left( \begin{array}{c}
\text{AGREEMENT} \\
\text{NUMBER} \ sg \\
\text{PERSON} \ 3^{rd}
\end{array} \right) \\
&\text{SUBJECT} = \left( \begin{array}{c}
\text{AGREEMENT} \\
\text{PERSON} \ 3^{rd}
\end{array} \right) \\
&\text{NUMBER} \ pl
\end{align*}
\]

\textbf{Fails!}
Unification of Feature Structure

\[
\begin{align*}
\text{AGREEMENT} & \quad \left( \text{NUMBER} \quad \text{sg} \right) \\
\text{SUBJECT} & \quad \left( \text{AGREEMENT} \quad \left( \text{NUMBER} \quad \text{sg} \right) \right) \\
\text{U} \quad \text{SUBJECT} & \quad \left( \text{AGREEMENT} \quad \left( \text{PERSON} \quad 3^{rd} \right) \right) \\
& \quad \left( \text{NUMBER} \quad \text{sg} \right) \right)
\end{align*}
\]
Unification of Feature Structure

\[
\begin{align*}
\text{U} & \begin{cases} 
\text{SUBJECT} \Rightarrow \text{AGREEMENT} \quad \text{NUMBER} \\ 
\text{SUBJECT} \Rightarrow \text{AGREEMENT} \quad \text{NUMBER} \\ 
\text{SUBJECT} \Rightarrow \text{AGREEMENT} \quad \text{PERSON} \quad 3^{rd} \\ 
\text{SUBJECT} \Rightarrow \text{AGREEMENT} \quad \text{NUMBER} \quad \text{sg} \\
\end{cases} \\
= & \begin{cases} 
\text{AGREEMENT} \\
\text{NUMBER} \\
\text{sg} \\
\text{PERSON} \quad 3^{rd} \\
\text{NUMBER} \quad \text{sg} \\
\end{cases}
\end{align*}
\]
Plan for the Talk

- Problems with CFG (PCFG)
- Features Structure
  - Attribute-value Matrix (AVM)
- Unification

Grammar formalisms based on unification
Grammar Theories based on Unification

- Lexical Functional Grammar (LFG) (Bresnan, 1982)
- Construction Grammar (Kay and Fillmore, 1999)
- Unification Categorial Grammar (Uszkoreit, 1986)

Note that these grammar formalisms tend to focus on illuminating syntactic analysis, rather than providing computational implementations. (computationally very expensive)
Example of AVM used in HPSG

  - Non-derivational
  - Constraint-based
  - Highly lexicalized

- Each word is fully described with
  - morpho-syntactic features
  - semantic features
Example of AVM used in HPSG

- “put” --- e.g., “John put a book on the table”
Example of AVM used in HPSG

• Each word can have many different AVM descriptions (due to polysemy, or multiple possible syntactic relations with other words/phrases)
• each lexical_entry corresponds to an AVM description such as shown below:

```
word \rightarrow \text{lexical\_entry\_1}
V \text{ lexical\_entry\_2}
V \ldots
V \text{ lexical\_entry\_n}
```
The Chomsky Hierarchy

- Regular (or Right Linear) Languages
- Context-Free Languages
- Mildly Context-Sensitive Languages
- Context-Sensitive Languages
- Recursively Enumerable Languages
The Chomsky Hierarchy

<table>
<thead>
<tr>
<th>Type</th>
<th>Common Name</th>
<th>Rule Skeleton</th>
<th>Linguistic Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Turing Equivalent</td>
<td>$\alpha \rightarrow \beta$, s.t. $\alpha \neq \epsilon$</td>
<td>HPSG, LFG, Minimalism</td>
</tr>
<tr>
<td>1</td>
<td>Context Sensitive</td>
<td>$\alpha A \beta \rightarrow \alpha \gamma \beta$, s.t. $\gamma \neq \epsilon$</td>
<td>TAG, CCG</td>
</tr>
<tr>
<td>-</td>
<td>Mildly Context Sensitive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Context Free</td>
<td>$A \rightarrow \gamma$</td>
<td>Phrase-Structure Grammars</td>
</tr>
<tr>
<td>3</td>
<td>Regular</td>
<td>$A \rightarrow xB$ or $A \rightarrow x$</td>
<td>Finite-State Automata</td>
</tr>
</tbody>
</table>

- Lexical Functional Grammar (LFG) (Bresnan, 1982)
- Minimalist Grammar (Stabler, 1997)
- Tree-Adjoining Grammars (TAG) (Joshi, 1985)
- Combinatory Categorial Grammars (CCG) (Steedman, 1996, 2000)
Turing Test

- **Turing Test**: Interrogator ‘c’ engages in a natural language conversation with ‘a’ and ‘b’ to determine which is a computer and which is a human.