Agenda

- HW4, due today!
- Questions, comments, concerns?
- Schedule changes on the syllabus
- Chomsky Hierarchy revisited
- Context-sensitive grammars
  - Unification
  - Tree-adjointing grammars (TAG)
  - Combinatory Categorial Grammars (CCG)

Chomsky Hierarchy

<table>
<thead>
<tr>
<th>Language</th>
<th>Mechanisms</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>Regular expressions</td>
<td>( xy^p ) Morphology</td>
</tr>
<tr>
<td></td>
<td>Regular grammars</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Finite-state automata</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WFSA/WFSTs</td>
<td></td>
</tr>
<tr>
<td>Context-free</td>
<td>Context-free grammars (CFGs)</td>
<td>( ab^n ) Most syntax</td>
</tr>
<tr>
<td></td>
<td>Pushdown automata</td>
<td></td>
</tr>
<tr>
<td>Context-sensitive</td>
<td>Unification grammars</td>
<td>( \alpha \beta \gamma \delta ) Cross-serial dependencies</td>
</tr>
<tr>
<td></td>
<td>Lexicalized formalisms (e.g., TAG, CCG)</td>
<td></td>
</tr>
</tbody>
</table>

Finite-State

Context-Free

Context-Sensitive: Unification

\[
S[\alpha] \rightarrow NP[\alpha] \ VP[\alpha] \\
NP[\alpha] \rightarrow DT \ NN[\alpha] \\
VP[\alpha] \rightarrow VB[\alpha] \ NP \\
the \ thieves \ see \ldots
\]
Unification: Feature Structures

\[
\begin{align*}
\text{FEATURE}_1 & : \text{VALUE}_1 \\
\text{FEATURE}_2 & : \text{VALUE}_2 \\
\ldots & \\
\text{FEATURE}_n & : \text{VALUE}_n 
\end{align*}
\]

e.g.,

\[
\begin{align*}
\text{CAT} & : \text{NP} \\
\text{NUMBER} & : \text{SG} \\
\text{PERSON} & : 3 
\end{align*}
\]

Feature Structures

• What do feature structures provide?
  • A mechanism to bring lexical features to bear on syntactic structure
  • A formal mechanism for handling how smaller constituents combine to form larger constituents
  • A mechanism to enforce constraints on syntactic structures, e.g.,
    • Agreement
    • Grammatical heads
    • Subcategorization
    • Long-distance dependencies

Feature Structures as Values

\[
\begin{align*}
\text{CAT} & : \text{NP} \\
\text{AGREEMENT} & : \text{NUMBER} \text{ SG} \\
\text{PERSON} & : 3 
\end{align*}
\]

\[
\begin{align*}
\text{CAT} & : S \\
\text{HEAD} & : \text{AGREEMENT} (1) \text{ NUMBER} \text{ SG} \\
\text{PERSON} & : 3 
\end{align*}
\]

Unification (\(\sqcup\))

• Unification (\(\sqcup\)) is an operation on feature sets
  • Matches in values succeed; mismatches fail
  • Feature values can be underspecified
  • Unification with an underspecified value forces a match, e.g.,
    \[
    \text{NUMBER} \text{ SG} \sqcup \text{NUMBER} = \text{NUMBER} \text{ SG}
    \]
  • Features not explicitly represented are underspecified
    \[
    \text{NUMBER} \text{ SG} \sqcup \text{NUMBER} = \text{NUMBER} \text{ SG}
    \]

Underspecification: Example

• Consider the noun “sheep”, which is either plural or singular
  • In other words, the category as a subject noun will be
    \[
    \text{SUBJECT} \left[ \text{AGREEMENT} \right]
    \]
  • Then plural verbs like are will force a plural unification, and singular verbs like is a singular unification

  e.g., The goshdern sheep are chasing my dog versus The goshdern sheep is chasing my dog

More complicated unification

\[
\begin{align*}
\text{AGREEMENT} (1) \text{ NUMBER} \text{ SG} \\
\text{SUBJECT} \text{ AGREEMENT} (1)
\end{align*}
\]

\[
\begin{align*}
\text{SUBJECT} \text{ AGREEMENT} (1) \text{ NUMBER} \text{ SG} \\
\text{PERSON} : 3 
\end{align*}
\]

\[
\begin{align*}
\text{SUBJECT} \text{ AGREEMENT} (1) \\
\text{NUMBER} : \text{SG} \\
\text{PERSON} : 3 
\end{align*}
\]
"Copying" via Unification

- What if we don’t yet know values, but know they should match?
  - e.g., an S node: NP and VP may be either singular or plural, but should definitely match

\[
\begin{align*}
\text{AGREEMENT} & \quad (1) \\
\text{SUBJECT} & \quad \text{AGREEMENT} \quad (1) \\
\cup & \\
\text{SUBJECT} & \quad \text{AGREEMENT} \quad \text{NUMBER} \quad \text{SG} \\
\quad & \\
\text{AGREEMENT} & \quad (1) \\
\text{SUBJECT} & \quad \text{AGREEMENT} \quad (1) \quad \text{NUMBER} \quad \text{SG} \\
\end{align*}
\]

Feature Structures

- What do feature structures provide?
  - A mechanism to bring lexical features to bear on syntactic structure
  - A formal mechanism for handling how smaller constituents combine to form larger constituents
  - A mechanism to enforce constraints on syntactic structures, e.g.,
    - Agreement
    - Grammatical heads
    - Subcategorization
    - Long-distance dependencies

Features Example: S-node Agreement

- S → NP VP

\[
\begin{align*}
\text{CAT} & \quad \text{S} \\
\text{HEAD} & \quad \text{AGREEMENT} \quad (1) \\
\text{SUBJECT} & \quad \text{AGREEMENT} \quad (1) \\
\uparrow & \\
\text{CAT} & \quad \text{NP} \quad \text{AGREEMENT} \quad (1) \\
\text{CAT} & \quad \text{VP} \quad \text{AGREEMENT} \quad (1)
\end{align*}
\]

Features: Heads

- Features for most categories are copied from one child, known as the head child

\[
\begin{align*}
\text{CAT} & \quad \text{NP} \quad \text{AGREEMENT} \quad (1) \\
\text{CAT} & \quad \text{DT} \quad \text{AGREEMENT} \quad (1) \\
\text{CAT} & \quad \text{Noun} \quad \text{AGREEMENT} \quad (3)
\end{align*}
\]

- Put AGREEMENT features under HEAD feature, and copy it all:

\[
\begin{align*}
\text{CAT} & \quad \text{NP} \quad \text{HEAD} \quad (1) \\
\text{CAT} & \quad \text{DT} \quad \text{AGREEMENT} \quad (2) \\
\text{CAT} & \quad \text{Noun} \quad \text{HEAD} \quad (1) \quad \text{AGREEMENT} \quad (2) \\
\text{CAT} & \quad \text{VP} \quad \text{HEAD} \quad (1) \quad \text{AGREEMENT} \quad (2) \\
\text{CAT} & \quad \text{Noun} \quad \text{AGREEMENT} \quad (3)
\end{align*}
\]
Head Constituents

- A common notion in both Linguistics and NLP is the head constituent, i.e., most important or driving constituent
- Example: in English, VP tends to be head of S
- Can define a recursive relation, down to lexical heads
- (S (NP The dog) (VP (VBD bit) (NP the mailman))):
  - the main verb is the head of the VP
  - the VP is the head of the S
  - thus “bit” is the lexical head of the S
  - final noun is typically considered head of NP (dog and mailman) although some linguists argue for the determiner to be head (DP)

Subcategorization

- Like the notion of a head child, subcategorization is a widespread idea
- Certain verbs require/allow certain arguments, e.g.,
  - give NP NP, give the library the book
  - give NP PP, give the book to the library
  - donate NP PP, donate the book to the library
  - * donate NP NP, donate the library the book
- These are syntactic constraints
- Semantic constraints are called selectional restrictions e.g., eat selects for edible objects
- “Fuzzier” restrictions, more easily violated

Subcategorization using Unification

\[
\begin{align*}
\text{CAT VP} & \rightarrow \text{CAT Verb HEAD (1) SUBCAT INTRANS} \\
\text{CAT VP} & \rightarrow \text{CAT Verb HEAD (1) SUBCAT TRANS} \rightarrow \text{CAT NP} \\
\text{CAT VP} & \rightarrow \text{CAT Verb HEAD (1) SUBCAT DITRANS} \rightarrow \text{CAT NP} \rightarrow \text{CAT NP}
\end{align*}
\]

Long-distance Dependencies

- Now that there are subcategorization constraints, a verb had better get its arguments
- What about the following: you give the book
  - No good as a stand-alone sentence (infinitive verb, missing an argument)
  - To which library did you give the book?
- Need some mechanism for allowing argument gaps
- These dependencies can be quite distant
  - Which flight do you want me to have the travel agent book?

Existing Unification Approaches

- Head-Driven Phrase Structure Grammar (HPSG) Pollard and Sag (1994)
- Feature structures have found their way into other approaches
- Ideas like head children and subcategorization are widespread
Agenda

• HW4, due today!
• Questions, comments, concerns?
• Schedule changes on the syllabus
• Chomsky Hierarchy revisited
• Context-sensitive grammars
  • Unification
  • Tree-adjoining grammars (TAG)
  • Combinatory Categorial Grammars (CCG)

Tree-adjoining Grammars (TAG)

• Initial, auxiliary and elementary trees
• Substitution and Adjunction
• Derived and derivation trees

Tree-adjoining Grammars

A Tree-adjoining grammar (TAG) \( G = (V, T, S^I, I, A) \)
- a set of non-terminal variables \( V \)
- a set of terminals \( T \)
- a special start symbol \( S^I \in V \)
- a set of initial trees \( I \)
  - Non-terminals on frontier marked for substitution
  - a set of auxiliary trees \( A \)
    - One non-terminal on frontier marked as foot node
    - Otherwise like initial trees

TAGs

• Elementary trees are of type \( X \) where \( X \) is the root category
• Foot node must be of same category as the root
• Lexicalized TAG (LTAG) requires at least one terminal item (the anchor) on every elementary tree
• Two operations defined on trees
  • Substitution
  • Adjunction
Adjunction (slide taken from Joshi & Schabes, 1997)

Derivation

• Derivation begins with a set of elementary trees like a set of rules in a CFG
• Trees can combine via substitution or adjunction
• Can define a “derives” relation, as with CFGs
• A string is in the “language” if there is a sequence of derives steps from the root symbol to a tree with the terminals at the frontier
• TAGs can generate cross-serial dependencies
• Derivation results in two trees:
  • derived tree
  • derivation trees

Context-Sensitive: Cross-serial Dependencies

Cross-serial Dependencies in TAG (slide taken from Joshi & Schabes, 1997)
Properties of TAGs

- TAGs are quite interesting formally, and a lot of work in formal language theory has been done
- The derivation trees are context-free, i.e. the derivation sequences form a context-free language
- There is a kind of pushdown automata that is weakly equivalent to TAGs
- Parsing complexity $O(n^3)$ compared to $O(n^5)$ for CFG

Agenda

- HW4, due today!
- Questions, comments, concerns?
- Schedule changes on the syllabus
- Chomsky Hierarchy revisited
- Context-sensitive grammars
  - Unification
  - Tree-adjointing grammars (TAG)
  - Combinatory Categorial Grammars (CCG)
- Homework 5 online tonight
End of lecture, 3 Nov.

Agenda
- HW4, due today!
- Questions, comments, concerns?
- Schedule changes on the syllabus
- Chomsky Hierarchy revisited
- Context-sensitive grammars
  - Unification
  - Tree-adjoining grammars (TAG)
  - Combinatory Categorial Grammars (CCG)

Categorial Grammars
- Approach has been around since the 50s (Bar-Hillel and Lambek)
- Closely tied to the Formal Semantics of Montague using lambda calculus
- TAG and CG have the same generative power: every TAG grammar has a weakly equivalent CG grammar (and vice versa)
- Notion of strong compositionality: syntactic structure and interpretation are derived in lockstep

Lexical Categories and Function Application
- Every word in the lexicon is associated with a complex grammatical category
- Two function application schemas, describing how to combine two categories to form a new category
  - X/Y Y \rightarrow X
  - Y X Y \rightarrow X
- Interpret X/Y as requiring a Y on the right to make an X
- Interpret X\Y as requiring a Y on the left to make an X
- Example: the is of category NP/N; big is N/N; car is N
- These categories are similar to elementary trees in TAG

CCG Example

Type-Lifting
- In function application there is a category that needs something and a category that fills that need
e.g., NP S/NP \rightarrow S
(S/NP needs an NP category on its left to give an S)
- One might also argue that an NP category needs an S/NP category on its right to give an S
- Something that needs an S/NP on its right to give an S is S(S/NP)
- Type lifting converts a category X to Y(X/Y) or Y(Y/X)
  for an arbitrary Y
Function Composition

- Consider John saw Mary
- John and Mary are of category NP
- saw is of category (S\NP)/NP
- We can type-lift John to S/(S\NP) giving:
  - John saw Mary
  - NP
  - (S\NP)/NP
  - S/(S\NP)
- S/(S\NP) needs something to its right that (S\NP)/NP will provide, once it gets an NP to its right
- Function composition allows these to combine as follows:
  - X/Y Y/Z \rightarrow X/Z or X/Y Z/X \rightarrow Z/Y

Compositional Semantics

- Great selling point: semantic categories associated with syntactic categories
- Lambda calculus provides natural formalism for deriving meaning of a constituent from the meaning of its children
- All operations discussed here have semantic correlate
  - Function application
  - Type lifting
  - Function composition

Lexicalized Grammar Formalisms

- TAG and CG are known as lexicalized grammars
- Have been shown to have weakly-equivalent generative capacity
- Lexical categories, not rules, specify how words combine
- Subcategorization is handled by the lexical categories of verbs
- Natural notion of lexical heads, also based on lexical categories
- Clear that many of the dependencies are lexical

Statistical Approaches

- As with unification grammars, each of these have had statistical approaches
- Some of the statistical approaches have involved finite-state approximations
  - “Supertagging” involves building a POS-tagger, with full TAG style lexical categories
  - Others involve log-linear models
  - Many statistical context-free parsing approaches are influenced by these formalisms and unification
  - Weighted, not categorical, constraints
Agenda: Summary

• HW4, due today!
• Questions, comments, concerns?
• Schedule changes on the syllabus
• Chomsky Hierarchy revisited
• Context-sensitive grammars
  • Unification
  • Tree-adjoining grammars (TAG)
  • Combinatory Categorial Grammars (CCG)
• Homework 5 online 11/3