Overview

• Motivation for lexical hierarchy
• Default inheritance
• Tour of the lexeme hierarchy
• The Case Constraint
• pos vs. lexeme
• Reading Questions
Motivation

• We've streamlined our grammar rules...
  • ...by stating some constraints as general principles
  • ...and locating lots of information in the lexicon.
  • Our lexical entries currently stipulate a lot of information that is common across many entries and should be stated only once.

• Examples?

• Ideally, particular lexical entries need only give phonological form, the semantic contribution, and any constraints truly idiosyncratic to the lexical entry.
Lexemes and Words

• **Lexeme**: An abstract proto-word which gives rise to genuine words. We refer to lexemes by their ‘dictionary form’, e.g. ‘the lexeme *run*’ or ‘the lexeme *dog*’.

• **Word**: A particular pairing of form and meaning. *Running* and *ran* are different words.
Lexical Types & Lexical Rules

• Lexemes capture the similarities among *run, runs, running, and run*.

• The lexical type hierarchy captures the similarities among *run, sleep, and laugh*, among those and other verbs like *devour* and *hand*, and among those and other words like *book*.

Q: What do *devour* and *book* have in common?
A: The SHAC

• Lexical rules capture the similarities among *runs, sleeps, devours, hands,*...
Default Inheritance

Q: Why do we have default inheritance?

A: Generalizations with exceptions are common:
  • Most nouns in English aren't marked for CASE, but pronouns are.
  • Most verbs in English only distinguish two agreement categories (3sing and non-3sing), but be distinguishes more.
  • Most prepositions in English are transitive, but here and there are intransitive.
  • Most nominal words in English are 3rd person, but some (all of them pronouns) are 1st or 2nd person.
  • Most proper nouns in English are singular, but some (mountain range names, sports team names) are plural.
Default Inheritance, Technicalities

If a type says \( \text{ARG-ST} / \langle \text{NP} \rangle \), and one of its subtypes says \( \text{ARG-ST} \langle \rangle \), then the ARG-ST value of instances of the subtype is \( \langle \rangle \).

If a type says \( \text{ARG-ST} \langle \text{NP} \rangle \), and one of its subtypes says \( \text{ARG-ST} \langle \rangle \), then this subtype can have no instances, since they would have to satisfy contradictory constraints.
Default Inheritance, More Technicalities

- If a type says MOD / < S >, and one of its subtypes says MOD <[SPR < NP>]>, then the ARG-ST value of instances of the subtype is what?

\[
\begin{align*}
\text{MOD} & \left\langle \begin{bmatrix}
\text{HEAD} & / \text{verb} \\
\text{SPR} & \langle \text{NP} \rangle \\
\text{COMPS} & / \langle \rangle
\end{bmatrix} \right\rangle
\end{align*}
\]

- That is, default constraints are ‘pushed down’
Question on Default Inheritance

Q: Can a grammar rule override a default constraint on a word?

A: No. Defaults are all ‘cached out’ in the lexicon.

- Words as used to build sentences have only inviolable constraints.
Our Lexeme Hierarchy

- synsem
  - [SYN, SEM]
  - lexeme
    - [ARG-ST]
      - expression
        - word
        - phrase
      - infl-lxm
        - const-lxm
          - pn-lxm
          - pron-lxm
          - adj-lxm
          - conj-lxm
          - det-lxm
          - predp-lxm
          - argmkp-lxm
        - verb-lxm
        - cn-lxm
          - siv-lxm
          - piv-lxm
          - tv-lxm
          - cntn-lxm
          - massn-lxm
        - stv-lxm
        - dtv-lxm
        - ptv-lxm

Functions of Types

• Stating what features are appropriate for what categories

• Stating generalizations
  • Constraints that apply to (almost) all instances
  • Generalizations about selection -- where instances of that type can appear
Every synsem has the features SYN and SEM

In the diagram:
- synsem [SYN, SEM]
- lexeme [ARG-ST]
- expression
  - word [ARG-ST]
  - phrase
- const-lxm
- infl-lxm
- adj-lxm
- conj-lxm
- det-lxm
- predp-lxm
- argmkp-lxm
- pn-lxm
- pron-lxm
- verb-lxm
- cn-lxm
- siv-lxm
- piv-lxm
- tv-lxm
- cntn-lxm
- massn-lxm
- stv-lxm
- dtv-lxm
- ptv-lxm

© 2003 CSLI Publications
No ARG-ST on phrase
A Constraint on *infl-lxm*: the SHAC
A Constraint on \textit{infl-lxm}: the SHAC

\[
\text{infl-lxm} : \begin{bmatrix}
\text{SYN} \\
\text{VAL} \\
\text{HEAD}
\end{bmatrix}
\begin{bmatrix}
\text{SPR} \left\langle \left[\text{AGR} \quad 1 \right] \right\rangle
\text{AGR} \quad 1
\end{bmatrix}
\]
Constraints on \textit{cn-lxm}
Constraints on *cn-lxm*

\[
\begin{align*}
\text{SYN} & \quad \text{VAL} & \quad \text{SEM} \\
\text{cn-lxm} : & \quad \begin{cases}
\text{HEAD} & \quad \text{noun} \\
\text{AGR} & \quad [\text{PER 3rd}] \\
\text{SPR} & \quad \langle \begin{cases}
\text{HEAD} & \quad \text{det} \\
\text{INDEX} & \quad i
\end{cases} \rangle
\end{cases} \\
\text{MODE} & \quad / \quad \text{ref} \\
\text{INDEX} & \quad i \\
\text{ARG-ST} & \quad \langle X \rangle \oplus / \langle \rangle
\end{align*}
\]
Formally Distinguishing Count vs. Mass Nouns

\[
\text{synsem} \quad [\text{SYN, SEM}]
\]

\[
\text{lexeme} \quad [\text{ARG-ST}]
\]

\[
\text{expression} \quad \text{word} \quad \text{phrase}
\]

\[
infl-lxm \quad \text{const-lxm}
\]

\[
\text{adj-lxm} \quad \text{conj-lxm} \quad \text{det-lxm} \quad \text{predp-lxm} \quad \text{argmkp-lxm}
\]

\[
\text{verb-lxm} \quad \text{cn-lxm}
\]

\[
\text{siv-lxm} \quad \text{piv-lxm} \quad \text{tv-lxm} \quad \text{cntn-lxm} \quad \text{massn-lxm}
\]

\[
\text{stv-lxm} \quad \text{dtv-lxm} \quad \text{ptv-lxm}
\]
Formally Distinguishing Count vs. Mass Nouns

cntn-lxm : \[
\begin{array}{c}
\text{SYN} \\
\text{VAL} \\
\text{SPR} \\
\langle [\text{COUNT} +] \rangle
\end{array}
\]

massn-lxm : \[
\begin{array}{c}
\text{SYN} \\
\text{VAL} \\
\text{SPR} \\
\langle [\text{COUNT} -] \rangle
\end{array}
\]
Constraints on *verb-lxm*

```
synsem
  [SYN, SEM]
  
lexeme
  [ARG-ST]
  
expression
  word
  phrase

infl-lxm
  
const-lxm
  

verb-lxm
  

cn-lxm
  

siv-lxm
 piv-lxm
 tv-lxm
 cntn-lxm
 massn-lxm

stv-lxm
 dtv-lxm
 ptv-lxm
```
Constraints on *verb-lxm*

\[
\text{verb-lxm: } \begin{bmatrix}
    \text{SYN} & \begin{bmatrix} \text{HEAD} & \text{verb} \end{bmatrix} \\
    \text{SEM} & \begin{bmatrix} \text{MODE} & \text{prop} \end{bmatrix} \\
    \text{ARG-ST} & / \langle \text{NP}, \ldots \rangle
\end{bmatrix}
\]
Subtypes of \textit{verb-lxm}:

- \textit{verb-lxm}: \([\text{ARG-ST} / < \text{NP, ...}>]\)
- \textit{siv-lxm}: \([\text{ARG-ST} / < \text{NP}>]\)
- \textit{piv-lxm}: \([\text{ARG-ST} / < \text{NP, PP}>]\)
- \textit{tv-lxm}: \([\text{ARG-ST} / < \text{NP, NP, ...} >]\)
- \textit{stv-lxm}: \([\text{ARG-ST} / < \text{NP, NP, } >]\)
- \textit{dtv-lxm}: \([\text{ARG-ST} / < \text{NP, NP, NP} >]\)
- \textit{ptv-lxm}: \([\text{ARG-ST} / < \text{NP, NP, PP} >]\)
Proper Nouns and Pronouns

\[
\begin{align*}
\text{pn-}l\text{xm:} & \quad \text{SYN} \begin{bmatrix}
\text{HEAD} \begin{bmatrix}
\text{noun} \\
\text{AGR} \begin{bmatrix}
\text{PER 3rd} \\
\text{NUM / sg}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix} \\
\text{SEM} \begin{bmatrix}
\text{MODE ref}
\end{bmatrix} \\
\text{ARG-ST} & \langle \rangle
\end{align*}
\]

\[
\begin{align*}
\text{pron-}l\text{xm:} & \quad \text{SYN} \begin{bmatrix}
\text{HEAD noun}
\end{bmatrix} \\
\text{SEM} \begin{bmatrix}
\text{MODE / ref}
\end{bmatrix} \\
\text{ARG-ST} & \langle \rangle
\end{align*}
\]
The Case Constraint

An outranked NP is [CASE acc].

- object of verb ✓
- second object of verb ✓
- object of argument-marking preposition ✓
- object of predicational preposition ✓

© 2003 CSLI Publications
The Case Constraint, continued

An outranked NP is [CASE acc].

- Subjects of verbs
  - Should we add a clause to cover nominative subjects?
    - No.

  *We expect them to leave.* (Chapter 12)

  - Lexical rules for finite verbs will handle nominative subjects.

- Any other instances of case marking in English?

- Does it apply to case systems in other languages?

  No: The Case Constraint is an English-specific constraint.
Apparent redundancy

• Why do we need both the pos subhierarchy and lexeme types?

• pos:
  • Applies to words and phrases; models relationship between then
  • Constrains which features are appropriate (no AUX on noun)

• lexeme:
  • Generalizations about combinations of constraints
Lexical Types & Lexical Rules

• Lexemes capture the similarities among *run*, *runs*, *running*, and *run*.

• The lexical type hierarchy captures the similarities among *run*, *sleep*, and *laugh*, among those and other verbs like *devour* and *hand*, and among those and other words like *book*.

• Lexical rules capture the similarities among *runs*, *sleeps*, *devours*, *hands*,...
Overview

• Motivation for lexical hierarchy
• Default inheritance
• Tour of the lexeme hierarchy
• The Case Constraint
• pos vs. lexeme
• Reading Questions
Reading Questions

• How do underspecification and defeasible constraints interact?

• Why do we have to specifically mark which constraints are defeasible?

• How do we know which ones are?

• Can other words/phrases in a sentence override defeasible constraints on some lexeme/word?

• Are there advantages to a model w/o defeasible constraints?
Reading Questions

• What do we mean by "object", "initial description", "final description", "lexical sequence", "family of lexical sequences"?

• How do lexemes fit into the grammar?

• What's the difference between "lexeme" and "lemma"?

• Why infinitely many lexical sequences for *dog*?
Reading Questions

• How do we handle verbs that do the dative-shift thing? Which category do we put them in?

• What's the difference between piv-lxm and stv-lxm? Or maybe we could have specified these as subchildren of some strict transitive type?

• Do predicative prepositions have ARG-ST? What about argument marking ones?

• What's with X and Y?
Reading Questions

• Which lexemes will override MOD /< >? 

• P. 243 says that only predicational prepositions can be modifiers. Are there any excepts to this or things that PPs can't modify?
Reading Questions

• Why didn't we just use multiple inheritance?

• Referring to the lexeme 'tree' on p233, can subtypes have subtypes? For example, if Tj branched off from Ti, can Tj have it's own subtypes as well? Could the subtypes of Tj have subtypes?

• Is there a way to formally acknowledge the crossover between pos and lexeme types?
Reading Questions

• The CASE Constraint looks like a hack. What's the point?

• With regards to the CASE constraint, what about a sentence like *Because of him, I lost the race.*?