

May 24, 2004

Long Distance Dependencies

Reflections on Grammar Engineering

Q&A

Overview

- Long distance dependencies: Bottom, middle and top
- Long distance dependencies demo
- What we've accomplished
- What we've needed to skirt
- What else there is to do...
- Grammar engineering
- Open Q&A

Long distance dependencies: Overview (1/4)

- In an LDD, a constituent appears to be ‘dislocated’ from its usual spot.

What did Kim see __ ?

Bagels, I like __ .

Presents from Grandma were hard for the children to discover __ .

Long distance dependencies: Overview (2/4)

- They are ‘long distance’, because there can be arbitrarily many clauses in between the ‘filler’ and the ‘gap’:

What did Sandy say Terry thought Pat believed

Kim saw __ ?

Bagels, I don't suppose I could ever convince you
to like __ .

Presents from Grandma are easy to help the
children discover __ .

Long distance dependencies: Overview (3/4)

- One approach is to describe LDDs in terms of movement, or the relationship between two different phrase structures.
- Alternatively, they can be viewed as a pattern of constraints on when an ‘extra’ constituent can be attached at the left edge of a clause:

What did they hand to the baby?

*What did they hand a toy to the baby?

*They handed to the baby.

Long distance dependencies: Overview (4/4)

- Some constituents are missing subconstituent (‘contain gaps’).
- Gap-containing constituents can be embedded in larger constituents, which either:
 - Also contain the filler for the gap.
 - Don’t, and are therefore also ‘gap-containing’.

Bottom, middle and top

- Bottom: Record the fact that something is missing.
 - Use a list-valued feature SLASH
 - Unary rules move elements from valence lists to SLASH, or introduce modifiers in SLASH
- Middle: Propagate the information that something is missing.
 - Heads collect SLASH values of their dependents.
 - Phrases collect SLASH values from their heads.
- Top: Pair a filler with the gap.
 - Phrase structure rules which require a slashed head daughter and a non-head daughter which matches the requirement encoded in the SLASH feature.

Adding non-subject wh questions to a matrix grammar

- Most of the work is already done in the matrix.
- For example, basic-one-arg etc types amalgamate slash values of dependents.
- Likewise, the matrix provides a basic-head-filler-phrase type, and an extracted-comp-phrase type, etc.
- What I had to add to my English grammar:
`http://courses/ling471/wh-english.txt`
- Demo...

What we've accomplished (1/2)

- Implementation of a basic word order
- Lexical classes of nouns, verbs (transitive, intransitive, ditransitive), determiners, adpositions
- Case (on core arguments, at least) and agreement (subject-verb, det-noun) as appropriate.
- Inflection marking (in)definiteness
- Adjectives/adverbs

What we've accomplished (2/2)

- Optionality of determiners and NP/PP arguments
- Sentential negation
- Modal semantics as a raising verb or verbal inflection
- Clausal syntax and semantics for matrix and embedded declarative and polar interrogative clauses.
- ...all in grammars which assign precise, elaborated semantic representations and can be used to parse and generate.

Things we explicitly didn't handle (1/2)

- Some morphoorthography (vowel harmony [Hungarian, Turkish], irregular stems [French, ...], interdigitated morphology [Arabic], morpheme-stripping rules [French, Hindi, Swedish, ...])
- Non-ascii orthographies
- Freer word order (V2 phenomena [Swedish], scrambling [Japanese], pragmatic constraints on word order [Hungarian])
- Serial verbs [Cantonese, Haitian Creole]

Things we explicitly didn't handle (2/2)

- Complex predicates [Farsi]
- Verb clusters [Hindi]
- Semantic selection involving classifiers [Cantonese, Navajo]
- Tense/aspect/mood semantics [All]
- ‘Particles’

Other topics which we didn't even get to

- Relative clauses
- Valence alternations (passive, anti-passive)...
- Coordination
- Politeness markers
- Imperatives
- ...

Reflections on Grammar Engineering (1/3)

- Grammar engineering requires linguistic analysis
- Three sources in this class:
 - The Matrix (top-down expectations)
 - Written sources
 - You!
- Grammars are always grammars of language fragments
 - There will always be more phenomena to handle
 - Often our first-pass analyses of the phenomena we do handle are incomplete
- Examples?

Reflections on Grammar Engineering (2/3)

- In an implemented grammar, analyses of varied phenomena must interact properly.
- This makes grammar engineering difficult...
- ...but it keeps up honest: varied phenomena coexist in natural language sentences, so adequate grammars should be able to handle them seamlessly.
- Regular regression testing is crucial.
- Examples?

Reflections on Grammar Engineering (3/3)

- From the first lecture, applications of precision grammars
 - language documentation/linguistic hypothesis testing
 - machine translation
 - automated email response
 - augmentative and assistive communication
 - computer assisted language learning
 - IR (from structured or unstructured data)
 - ...
- What are the advantages and disadvantages to engineering one grammar for these diverse goals?

Reflections on the Grammar Matrix (1/2)

- One of the goals of Grammar Matrix development is a bottom-up exploration of cross-linguistic universals.
- We believe that the formalism is sufficiently powerful to allow observationally, descriptively, and explanatorily adequate analyses (i.e., get the grammaticality judgments right, get the semantics right, capture generalizations).
- In addition, we are proposing some initial hypotheses about language universals in the Grammar Matrix.

Reflections on the Grammar Matrix (2/2)

- Both of these resources (formalism and matrix) provide some top-down expectations about languages, and therefore must be used carefully.
- Examples of cases where the expectations seemed a poor fit for your language?

Overview

- Long distance dependencies: Bottom, middle and top
- Long distance dependencies demo
- What we've accomplished
- What we've needed to skirt
- What else there is to do...
- Grammar engineering
- Open Q&A