Knowledge Engineering for NLP

February 5, 2007

Discourse status

Optional arguments, Modification,

Overview

- Discourse status
- Optional arguments
  - Semantic classification
  - Syntactic classification
  - Typological claims
- Analysis of optional arguments
- Modification

Discourse status: What’s that? (1/2

- A property of referents, describing their relationship to the common ground of a conversation.
- Tends to be reflected syntactically in markers of ‘definiteness’ as well as demonstratives and constraints on the availability of types of NPs in particular constructions.
- Closely related to information structure:
  - Classification parts of a sentence into topic and comment
  - Sentential focus

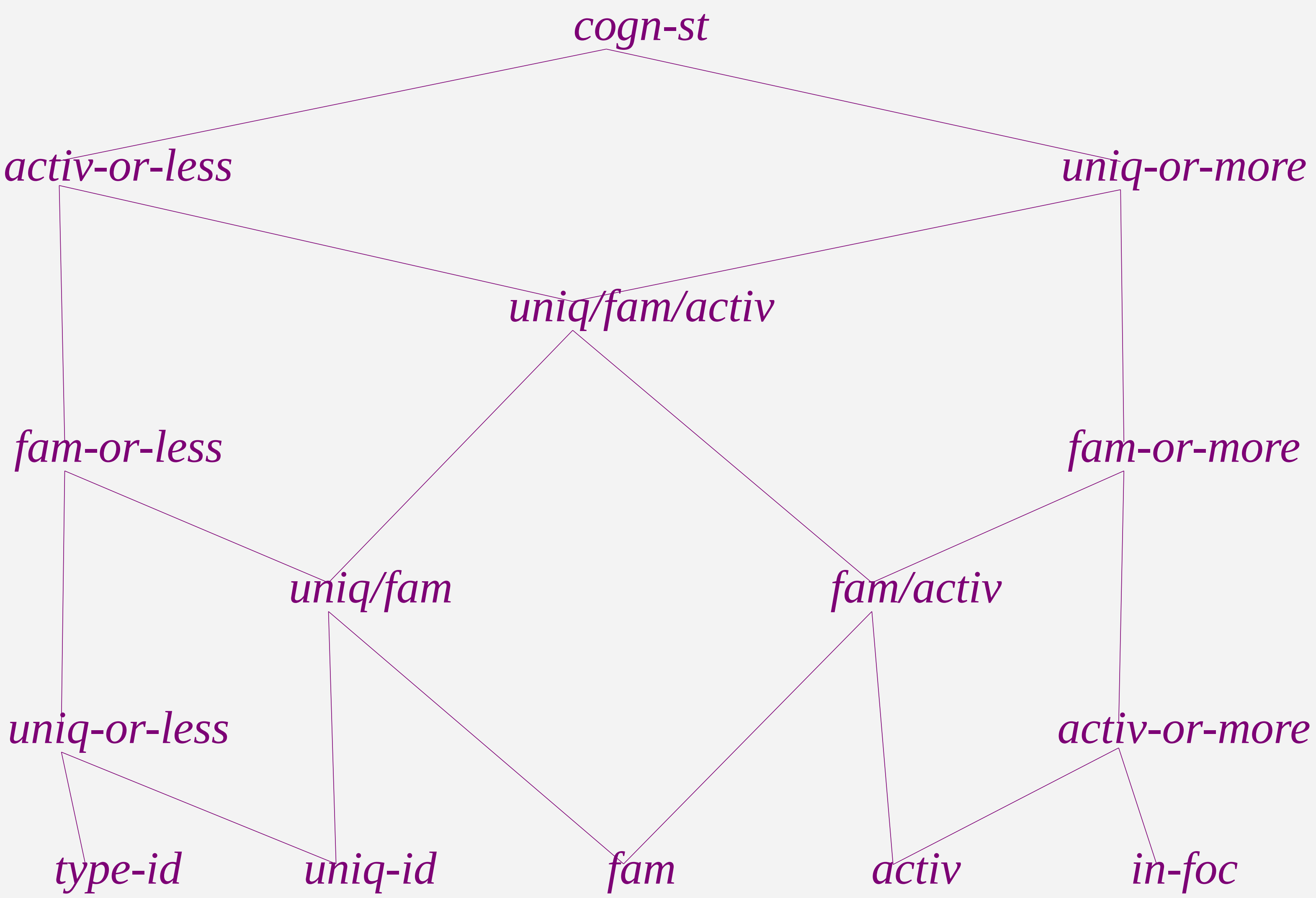
Discourse status: What's that? (2/2)

- The binary disctinction "definite/indefinite" is not sufficient to capture this.
- Furthermore, discourse status can be broken down into hearer-oriented "cognitive status" and speaker-oriented "specificity."

Givenness hierarchy (Gundel et al 1993, Prince 1981)

In focus>	Activated>	Familiar>	Uniq. id.>	Referential>	Type id.
it	*that, this* *this N*	*that N*	*the N*	indefinite *this N*	*a N*

Borthen & Haugereid’s proposal (1/3)



Borthen & Haugereid's proposal (2/3)

$$\begin{bmatrix} \text{SS.LOC.CONT.REF-PROP} & \begin{bmatrix} \textit{ref-prop} & \\ \text{INDEX} & \begin{bmatrix} \textit{ref} & \\ \text{PER} & \textit{per} \\ \text{NUM} & \textit{num} \\ \text{GEND} & \textit{gend} \end{bmatrix} \\ \text{COGN-ST} & \textit{cogn-st} \\ \text{SPECI} & \textit{bool} \\ \text{PART} & \textit{bool} \\ \text{UNIV} & \textit{bool} \end{bmatrix} \end{bmatrix}$$

Borthen & Haugereid's proposal (3/3)

- SPECI indicates specificity (speaker-oriented)
- Compatible with both "definite" and "indefinite" NPs:
  - *The best student won.*
  - *The next customer will receive a reward.*
- Corresponds to overt syntactic phenomena in at least Norwegian (specificity adjectives) and Turkish (accusative case precluses specific interpretation).

First-pass Matrix-based proposal

$$\begin{bmatrix} \text{HOOK.INDEX} & \begin{bmatrix} \text{PNG} & \begin{bmatrix} \text{PER} & \textit{person} \\ \text{NUM} & \textit{number} \\ \text{GEND} & \textit{gender} \end{bmatrix} \\ \text{COG-ST} & \begin{bmatrix} \textit{cog-st} & \\ \text{SPECI} & \textit{bool} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Optional arguments

- There are many cases in which an argument may be semantically present but syntactically absent.
- Semantically, these cases can be categorized by how the missing argument is interpreted.
- Syntactically, these cases can be categorized by how the missing argument is licensed.

Semantic classification

- Indefinite null instantiation: *I ate.*

  The referent of the missing argument is indefinite, not (necessarily) recoverable from context.

- Definite null instantiation: *I told you already.*

  The referent of the missing argument is definite, i.e., it should be recoverable from context.

- Constructional null instantiation: *Eat!*, *I told Kim to eat*

  The referent of the missing argument is determined by the syntactic construction.

Syntactic classification

- Lexical: The potential for an argument to be missing is determined by the lexical type/entry of the selecting head.
    - *eat* allows indefinite null instantiation of its object
    - *devour* does not.
- Systematic: Arguments (perhaps of a certain syntactic type, such as NP or a particular grammatical function) in general can be missing.
    - Japanese-style any argument pro-drop
    - Spanish-style subject pro-drop.

Syntactic classification (2/2)

- By hypothesis, systematic pro-drop is given the definite interpretation (i.e., it corresponds to one use of overt pronouns in other languages).
- Pronoun incorporation: Verbal affixes are actually interpreted as pronouns. I would expect these cases to involve definite null instantiation.

Lining up syntactic and semantic classifications

- Claim 1: A language with systematic pro-drop will allow definite interpretations of all dropped arguments.
- Claim 2: A language with systematic pro-drop will also allow indefinite interpretations of some dropped arguments, corresponding roughly to where a language without systematic pro-drop would allow indefinite null instantiation.
- Claim 3: Indefinite null instantiation of subjects involves special verb marking (e.g., impersonal passives).
- Claim 4: It follows from these hypotheses that there is no need for lexically licensed definite null instantiation in languages with Japanese-style pro-drop.

Example (Japanese)

Tabeta

Ate

‘I/you/he... ate.’/‘I/you/he... ate it.’

- Japanese has systematic pro-drop of all arguments.
- It also appears to have lexically licensed INI.
- Thus *Tabeta* is ambiguous, and we would like to be able to translate it into two different English strings.
- Nonetheless, it would be nice to avoid assigning two different tree structures, and rather provide an underspecified semantic representation.

Proposed analysis in the Matrix: Overview (1/2)

- Constructional null instantiation covered by analysis of imperatives, raising, etc.
- Distinction between definite and indefinite null instantiation handled by a feature on indices representing definiteness.
  - Pronouns, arguments subject to DNI (and possibly definite NPs) are [COG-ST fam-or-more & [SPECI +]].
  - Arguments subject to INI (and possibly indefinite NPs) are [COG-ST type-id & [SPECI –]].
- Caveat: I'm not quite sure yet how to implement the cognitive status information, nor how it aligns with this distinction.

Proposed analysis in the Matrix: Overview (2/2)

- Posit opt-comp and opt-subj rules parallel to the bare-np rules.
- Use a feature [OPT bool] to code lexically licensed null instantiation (leaving it underspecified in languages where there is systematic pro-drop).
- Use a second feature [OPT-CS cog-st] to allow lexical items to specify whether any given optional argument would be interpreted as definite or indefinite in case of null instantiation. (As a stand-in for a semantic-interface based approach.)

The feature OPT

- OPT and OPT-CS will both be features of *synsems*.
- However, nothing constrains its own OPT value (that is, no phrases are inherently optional or non-optional, independent of which head they are dependent on).
- Rather, heads constrain certain arguments to be [OPT −], which blocks the optional complement/subject rules from applying, since these look for argument which are (compatible with) [OPT +].

The feature OPT-CS (1/2)

- OPT-CS is a ‘junk slot’ to allow a lexical head to store information about how an argument will be interpreted if it is unexpressed.
- The opt-comp rule will identify the OPT-CS and HOOK.INDEX.COG-ST values of any argument it caches out as unrealized.

The feature OPT-CS (2/2)

- Because the HOOK.INDEX of every argument is identified with some ARGn position in the head’s key relation, this information will be encoded in the semantics.
- Note that we’re not positing pronoun relations or associated quantifier relations for these dropped objects. This point is debatable, especially if your language appears to have incorporated pronouns.

The Matrix opt-comp type

```
basic-head-opt-comp-phrase := head-valence-phrase & head-only &
                              head-compositional &
  [ INFLECTED #infl,
    SYNSEM canonical-synsem &
    [ ..CAT [ VAL [ SUBJ #subj, COMPS #comps, SPR #spr, SPEC #spec ],
              MC #mc, POSTHEAD #ph ],
      MODIFIED #mod ],
    HEAD-DTR [ INFLECTED #infl & +,
               ..CAT [ VAL [ SUBJ #subj, SPR #spr, SPEC #spec,
                             COMPS < unexpressed &
                                   [ OPT +, OPT-CS #def,
                                     ..INDEX.COG-ST #def ] . #comps >],
                                     MC #mc, POSTHEAD #ph ],
               ..CONT.HOOK.INDEX event,
               MODIFIED #mod ],
    C-CONT [ RELS <! !>, HCONS <! !> ] ].
```

For a language with systematic pro-drop

- Allow definite null instantiation (pro-drop) everywhere.
- Also allow indefinite null instantiation if lexically specified.
- Same head-opt-comp-rule
- Two types of lexical entry:
  - Those that allow both INI and DNI leave OPT-CS undespecified
  - Those that only allow DNI specify [OPT-CS activ-or-more]

For Lab 6 (1/2)

- Determine whether your language allows systematic pro-drop, and if so, under what conditions (subjects only, all arguments, nearly all arguments, complements of verbs but not of adpositions, ...)
- Determine whether your language allows indefinite null instantiation for the objects of any verbs in your lexicon (*eat* would be a good guess).
- Determine whether your language has incorporated pronouns.

For Lab 6 (2/2)

- If your language doesn't allow pro-drop everywhere, determine whether it nonetheless allows lexically licensed definite null instantiation.
- Try to find out whether your language allows indefinite null instantiation of subjects (whether or not it's a pro-drop language). Good places to look are translations of *There was dancing at the party*, and similar.

Modification: Syntax

- Modifiers select the heads they modify via the MOD feature (inside HEAD).
- The value of MOD is a list of *synsems*.
- Head-modifier rules are cross-classified according to order (head-adj, adj-head) and the intersective/scopal distinction.
- You might already have head-modifier rules in your grammar (probably just instances in rules.tdl which inherit directly from types in matrix.tdl).

Intersective modifiers

- Adjoined via a ‘head-compositional’ PSR (syntactic head is semantic head)
- ARG1 is MOD’s INDEX (*individual*)
- LTOP = MOD’s LTOP (constraint on rule)

Scopal modifiers

- Serve as semantic head daughters

  What does this mean in tdl?

- Identify their own INDEX with their MOD’s INDEX (why?)
- Take a handle-valued ARG1
- Insert a qeq between their ARG1 and their MOD’s LTOP (why?)

Scopal modifiers: examples

- Kim did not read every book.
- Kim probably read every book.
- The most likely winner of every medal was disqualified.

Other non-intersective modifiers

- The alleged criminal
- The fake gun
- . . .

Gate keeping

- The phrase structure rules for intersective and scopal modifiers need to be different.
- Ponder why (‘an apparently difficult problem’)
- Use subtypes of *local* to constrain which rule gets used.

  No other use for subtypes of *local*

  Modifiers constrain LOCAL inside their MOD value

Scopal mod phrase

```
scopal-mod-phrase := head-mod-phrase-simple &
  [ NON-HEAD-DTR.SYNSEM.LOCAL [
       CAT.HEAD.MOD < [ LOCAL scopal-mod ] >,
       CONT.HOOK #hook ],
    C-CONT [ HOOK #hook,
             HCONS <! !> ] ].
```

Intersective mod phrase

```
isect-mod-phrase := head-mod-phrase-simple &
                    head-compositional &
  [ HEAD-DTR.SYNSEM.LOCAL.CONT [
        HOOK.LTOP #hand,
        MSG no-msg ],
    NON-HEAD-DTR.SYNSEM.LOCAL [
        CAT.HEAD.MOD < [ LOCAL intersective-mod ] >,
CONT.HOOK.LTOP #hand ],
    C-CONT.HCONS <! !> ].
```

Open issues

- Possible positions for adverbs (of different classes)
- Semantically, should *fake* and *likely* get the same treatment?
- Non-iterating modifiers (though we've made some progress this quarter)
- Allowing heads to be sensitive to properties of modifiers (e.g., ADV-aa in Kannada questions)

Overview

- Optional arguments
  - Semantic classification
  - Syntactic classification
  - Typological claims
- Analysis of optional arguments
- Modification