Grammar Engineering

April 18, 2005

Optional arguments, MRS

Overview

- Optional arguments
- Analysis of optional arguments
- How the LKB applies lexical rules
- MRS: representations
- (Next time) MRS: composition

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: No language allows indefinite null instantiation of subjects. [I rather expect this one to be falsified, cf. German 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 [DEF bool].
  - Pronouns, arguments subject to DNI (and possibly definite NPs) are INDEX.DEF +.
  - Arguments subject to INI (and possibly indefinite NPs) are INDEX.DEF -.

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-DEF bool] 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 DEF-OPT 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 DEF-OPT (1/2)

- DEF-OPT 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 DEF-OPT and HOOK.INDEX.DEF values of any argument it caches out as unrealized.

The feature DEF-OPT (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 +, DEF-OPT #def,
                                     ..INDEX.DEF #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.
- Two rules:
  - One head-opt-comp rule doesn't look at DEF-OPT value and just puts in DEF +.
  - The other requires [DEF-OPT –] and copies that to DEF.
- Lexical items specify [DEF-OPT +] if they don't allow indefinite null instantiation (of that argument).

For Lab 4 (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 whehter your language has incorporated pronouns.

For Lab 4 (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.

Semantics: Overview

- What we’re trying to represent
- Motivation and specification of representations
- Building up the representations compositionally (next time)

Semantics: Overall strategy

- Represent all semantic distinctions which are syntactically (or morphologically) marked.
- Underspecify semantic distinctions which don’t correspond to differences in form.
- (These can be ‘spelled out’ in post-processing.)
- Abstract away from non-semantic information (case, word order)
- Aim for consistency across languages (for purposes of downstream processing).
- Allow for semantic differences between languages.

Semantics: Scope

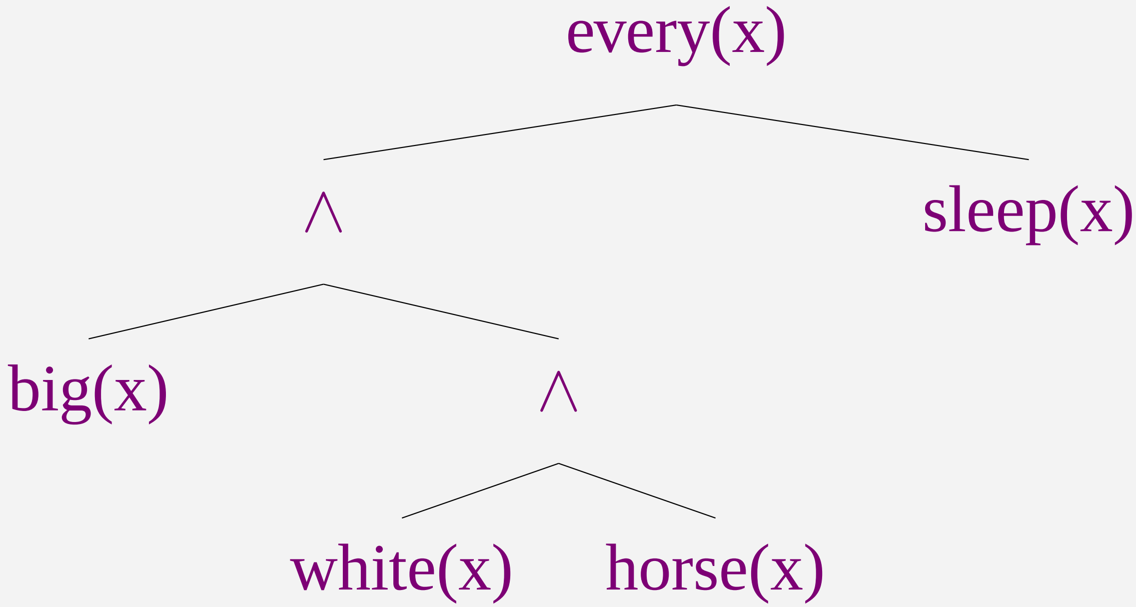
- Quantifiers (predicate logic or natural language) take three arguments:
  - A variable to bind
  - A restriction
  - A body
- Every dog sleeps: $\forall x\, dog(x) sleep(x)$
- When one quantifier appears within the restriction or body of another, we say the first has wider scope.

MRS: Goals

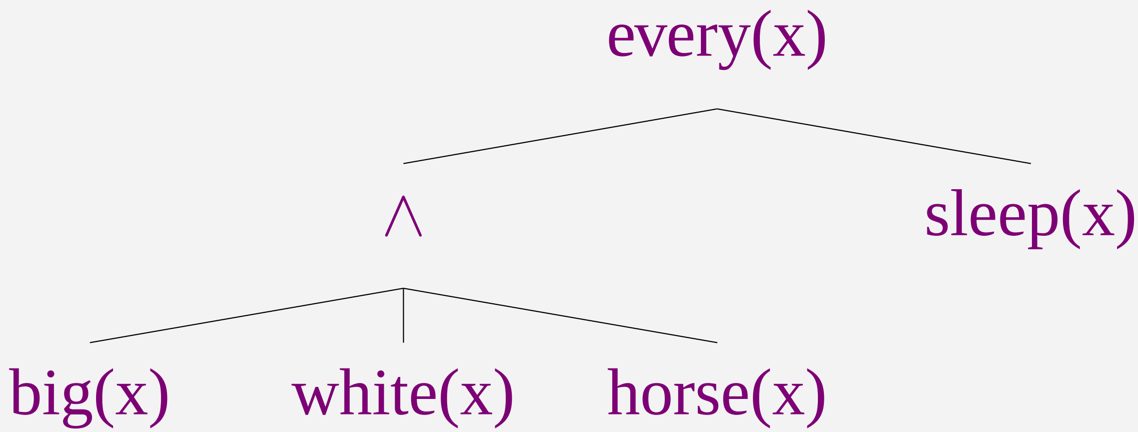
- Adequate representation of natural language semantics
- Grammatical compability
- Computaitonal tractability
- Underspecifiability

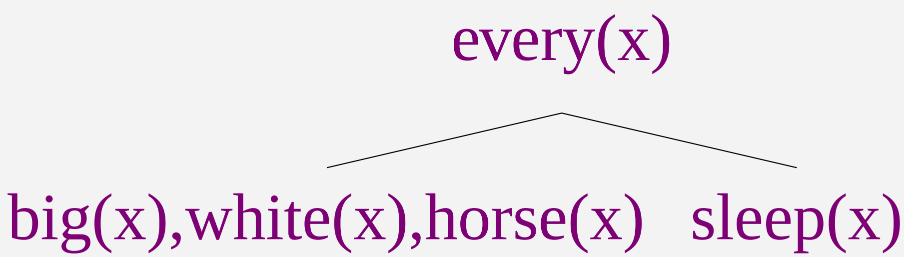
Working towards MRS (1/4)

- Every big white horse sleeps.
- every (x, $\wedge$(big(x), $\wedge$(white(x),horse(x))), sleep(x))



Working towards MRS (2/4)





Working towards MRS (3/4)

h0:every(x)

h1 h2

h1:big(x), h1:white(x), h1:horse(x) h2:sleep(x)

- And finally:

  $h0$:every(x, $h1$, $h2$), $h1$:big(x), $h1$:white(x), $h1$:horse(x), $h2$:sleep(x)

Working towards MRS (4/4)

- This is a flat representation, which is a good start.
- Next we need to underspecify quantifier scope, and it’s easier to see why with multiple quantifiers.
- At the same time, we want to be able to partially specify it, since this is required for adequate representations of NL semantics.

Underspecified quantifier scope (1/2)

- Every dog chases some white cat.

- some(y)
  - white(y),cat(y)
  - every(x)
    - dog(x)
    - chase(x,y)

- every(x)
  - dog(x)
  - some(y)
    - white(y),cat(y)
    - chase(x,y)

Underspecified quantifier scope (2/2)

- $h1$:every(x,$h3$,$h4$), $h3$:dog(x), $h7$:white(y), $h7$:cat(y), $h5$:some(y,$h7$,$h1$), $h4$:chase(x,y)
- $h1$:every(x,$h3$,$h5$), $h3$:dog(x), $h7$:white(y), $h7$:cat(y), $h5$:some(y,$h7$,$h4$), $h4$:chase(x,y)
- $h1$:every(x,$h3$,hA), $h3$:dog(x), $h7$:white(y), $h7$:cat(y), $h5$:some(y,$h7$,hB), $h4$:chase(x,y)

Partially constrained quantifier scope (1/5)

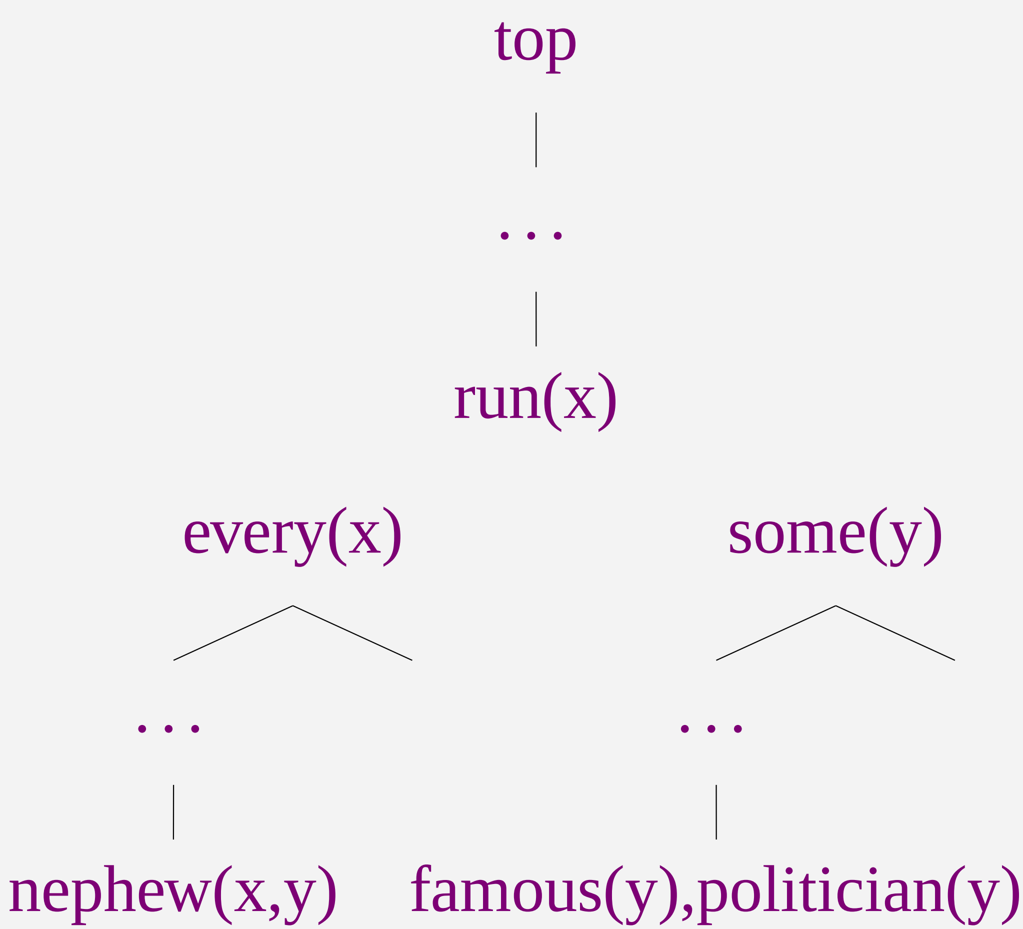
- For the BODY of quantifiers, we have no particular constraints to add.
- In turns out that the RESTRICTION needs to have partially underconstrained scope:
  - Every nephew of some famous politican runs.
  - every(x,some(y,famous(y) $\wedge$ politican(y), nephew(x,y)) run(x))
  - some(y,famous(y) $\wedge$ politican(y), every(x, newphew(x,y),run(x)))
  - But not:

- every(x,run(x),some(y,famous(y) $\wedge$ polician(y), nephew(x,y)))
- ‘Everyone who runs is a newphew of a famous politician.’

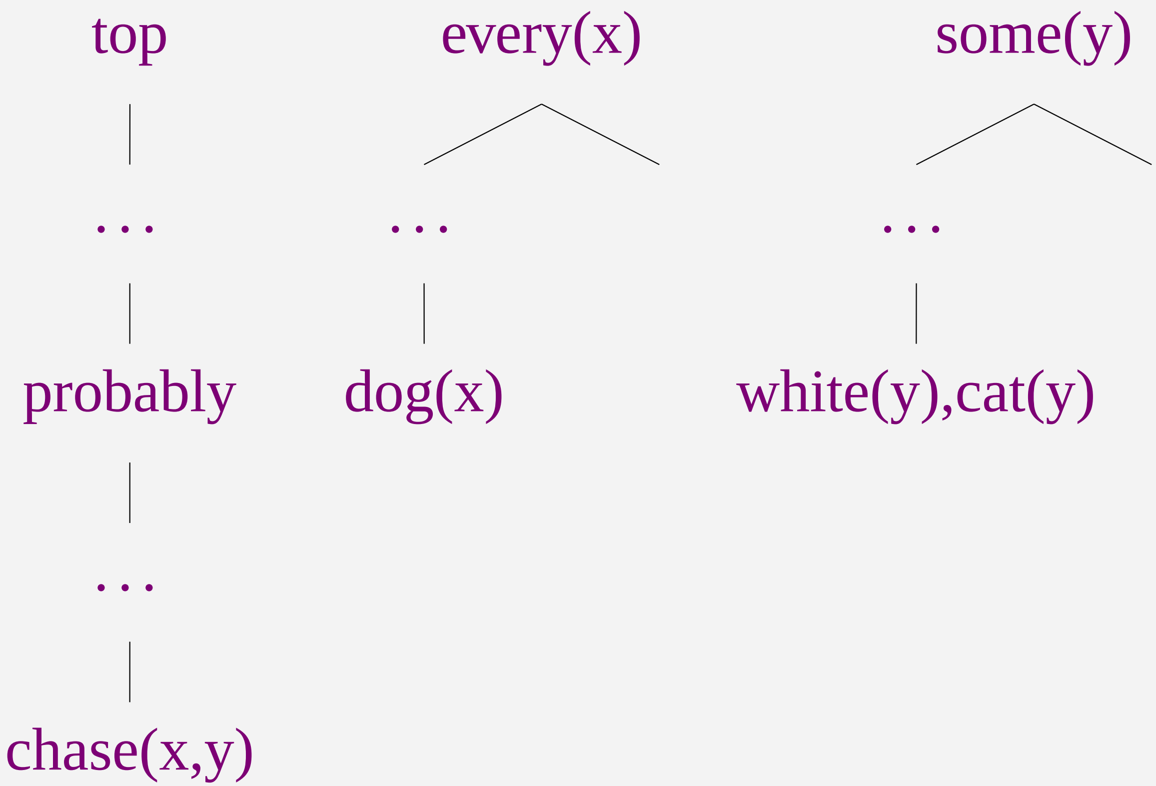
Partially constrained quantifier scope (2/5)

- Every nephew of some famous politican runs.
  - But not:
  - every(x,run(x),some(y,famous(y) $\wedge$ polician(y), nephew(x,y)))
  - 'Everyone who runs is a newphew of a famous politician.'

Partially constrained quantifier scope (3/5)



Partially constrained quantifier scope (4/5)



Partially constrained quantifier scope (5/5)

- $\langle h0, \{h2 : \text{every}(x, h3, h4), h5 : \text{nephew}(x, y), h6 : \text{some}(y, h7, h8), h9 : \text{politician}(y), h9 : \text{famous}(y), h10 : \text{run}(x)\}, \{h1 =_q h10, h7 =_q h9, h3 =_q h5\}\rangle$
- $\langle h0, \{h1 : \text{every}(x, h2, h3), h4 : \text{dog}(x), h5 : \text{probably}(h6), h7 : \text{chase}(x, y), h8 : \text{some}(y, h9, h10), h11 : \text{white}(y), h11 : \text{cat}(y)\}, \{h0 =_q h5, hw =_q h4, h6 =_q h7, h9 =_q h11\}\rangle$

We've arrived at MRS!

- Flat structure
- Underspecification/partial specification of scope is possible

Linguistic questions

- How do we build MRS representations compositionally?
- Is it linguistically adequate to insist that no process suppress relations?
- Under what circumstances do NLs (partially constrain scope)?
- Is it linguistically adequate to give scopal elements (esp. quantifiers, but also scopal modifiers) center-stage?

MRS in feature structures

- RELS: List (diff-list) of relations
- HCONS: List (diff-list) of handle constraints
- HOOK: Collection of features ‘published’ for further compisition: INDEX, LTOP, XARG
- ARGn: Roles within relations

Overview

- Optional arguments
- Analysis of optional arguments
- How the LKB applies lexical rules
- MRS: representations
- (Next time) MRS: composition