April 19, 2004

Case, agreement, modification and lexical rules

Overview

- Some things I ought to mention
- Case
- Agreement
- Modification
- Lexical rules
- Morpho-orthography

Some things I ought to mention

- Pep talk: This class and universal grammar
- There’s no magic here...

Case: Correlating noun form with grammatical function

- Not relevant in all languages
- Handle via a feature CASE on feature structures of type *noun*
- Verbs (and prepositions ...) indicate the CASE value they require for each nominal dependent
- In other words, case is a property of nouns that is selected by verbs (and prepositions, and other nouns, and ...)

Case: An example (1/2)

- Japanese:

  Kaze-ga mado-wo akeru

  Wind-NOM window-ACC open

  'The wind opens the window.

- Type definitions:

```
noun := head &
  [ CASE case ].

case := *top*.  acc := case.
nom := case.    dat := case.
```

Case: An example (2/2)

- Nouns:

```
kaze-ga := noun-lex &
  [ STEM < "kaze","ga" >,
    SYNSEM [ LOCAL.CAT.HEAD.CASE nom,
             LKEYS.KEYREL _wind_n_rel ] ].
```

- Verbs (add the following to the usual):

```
trans-verb-lex := verb-lex &
  [ ...VAL [ SUBJ < [ ...HEAD.CASE nom ] >,
             COMPS < [ ...HEAD.CASE acc ] > ]].
```

Agreement

- Agreement is when grammatical properties of one word/phrase are reflected in the morphology of another.
- What kinds of properties are so reflected?
- What kinds of words can do the reflecting?

AGR and png (1/2)

- Person, number and gender are encoded in the value of PNG, which is a feature of referential indices.
- (This is following Pollard & Sag's semantic treatment of agreement.)
- Even in languages without person/number agreement, we'll want to provide person and number information for semantic reasons.

AGR and png (2/2)

- The Matrix defines the value of PNG as *png*, but gives neither features appropriate for the type nor subtypes, as these are taken to be language-specific.
- Elements which agree with a noun in person, number or gender all have access to that noun's index, and thus can see this information.
- Verbs, adjectives, determiners thus don't necessarily have their own person/number/gender value but rather constrain the PNG of nouns they combine with.
- The Matrix provides another feature AGR, but that's an advanced topic we'll save for another time...

Subject-verb agreement: Example (1/2)

- French: Le chat éternue

  The cat sneezes-3SG

- Type definitions:

```
png-fr := png &
  [ PER person,
    NUM number,
    GEND gender ].

person := *top*.
third := person.
...
```

Subject-verb agreement: Example (2/2)

```
chat := noun-lex &
[ STEM < "chat" >,
  SYNSEM [ ...HOOK.INDEX.PNG [ PER third,
                               NUM sg,
                               GEND masc ],
           LKEYS.KEYREL _cat_n_rel ] ].

eternue := intrans-verb-lex &
[ STEM < "eternue" >,
  SYNSEM [ ...SUBJ < [ ...PNG [ PER third,
                                NUM sg ] ] >,
           LKEYS.KEYREL _sneeze_v_rel ] ].
```

Determiner-noun agreement: Example (1/2)

- Spanish:

  el gato estornuda

  the-SG.MASC cat sneeze-3SG

  'The cat sneezes'

- Types:

```
png-sp := png &      number := *top*.
  [ PER person,      sg := number.
    NUM number,      gender := *top*.
    GEND gender ].   masc := gender.
...
```

Determiner-noun agreement: Example (2/2)

```
gato := noun-lex &
   [ STEM < "gato" >,
     SYNSEM [ ...INDEX.PNG [ PER third,
                             NUM sg,
                             GEND masc ],
              LKEYS.KEYREL _cat_n_rel ] ].

el := det-lex &
   [ STEM < "el" >,
     SYNSEM [ ...SPEC < [ ...PNG [ NUM sg,
                                   GEND masc ] ] >,
              LKEYS.KEYREL def_q_rel ] ].
```

Modification

- Different modifiers go with different heads. (Examples?)
- Modifiers need access to semantic information about heads. (Why?)
- Propose a feature MOD, similar to the valence features, but inside HEAD.
- Head-modifier-rules match the MOD value of the non-head (modifier) daughter to the SYNSEM value of the head daughter.
- The matrix distinguishes scopal from intersective modification so we can get the facts right about things like *the apparently fake gun*, but that's beyond the scope of this class.

Adjective-noun agreement (1/2)

- Arabic: alhirr alkabirr 3aTas

  the-cat the-big sneezed

- From the orthography, guess that the definite article is actually an inflection on the noun.

```
affix-det-lex-rule := infl-ltol-rule &
  [ SYNSEM.LOCAL.CAT [ VAL.SPR < >,
                       HEAD.DEF + ],
    DTR.SYNSEM.LOCAL [ CAT.VAL.SPR < [...HEAD det] >,
                       CONT.HOOK [ INDEX #ind,
                                   LTOP #larg ] ],
    C-CONT [ RELS <! quant-relation &
                     [ PRED def_q_rel,
                       ARG0 #ind, RSTR #harg ] !>,
             HCONS <! qeq &
                     [ HARG #harg, LARG #larg ] !> ] ].


affix-det :=
%prefix (* al)
affix-det-lex-rule.
```

Adjective-noun agreement (2/2)

- Arabic: alhirr alkabirr 3aTas

  the-cat the-big sneezed

```
noun-lex := basic-noun-lex & ...
[ SYNSEM.LOCAL.CAT.HEAD.DEF - ].

alkabiir := adjective-lex &
[ STEM < >,
  SYNSEM [ ...HEAD.MOD <[...HEAD.DEF + ]>,
           LKEYS.KEYREL _big_j_rel ]].
```

Lexical rules

- Reduce redundancy:
  - One lexical entry per lemma
  - One lexical rule per inflection
- Useful for:
  - Case
  - Agreement
  - Tense
  - Voice alternations & other valence (diathesis) alternations
  - Nominalization

Lexical rules: Conceptual description

- Unary branching rules
- Daughter is “input”
- Mother is “output”
- May or may not change the orthography
- Typically, mother and daughter share a lot of information, changing on a little here and there
- Lexical rules may add semantic relations; may not take any away

Lexemes v. words

- Posit a feature INFLECTED appropriate for *sign*s.
- In languages with inflection, most lexical entries are [INFLECTED –] (i.e. ‘lexemes’).
- Through lexical rules, these give rise to families of related [INFLECTED +] forms (i.e., ‘words’).
- Nonetheless, some words can be lexically [INFLECTED +].
- Phrase structure rules require [INFLECTED +] daughters.

Cross-classification of lexical rules

- lexeme-to-lexeme v. lexeme-to-word:
  - ltol: Output still requires further inflection, rule can make arbitrary changes to SYNSEM
  - ltow: Output is ready to go out into the syntax, rule can only add information to SYNSEM
- constant-lex-rule v. inflecting-lex-rule:
  - constant: Input and output have the same STEM, identified to the LKB via a feature [NEEDS-AFFIX +]
  - inflecting: Input and output have different STEM

Non-"Morphological" rules: Example

- Ex: English dative alternation:

  Kim gave Sandy a book

  Kim gave a book to Sandy

- Define types *ditrans-verb-lex* and *pptrans-verb-lex* for each of these.

- Give lexical entries for just one *ditrans-verb-lex* and use lexical rule to derive the other.

Non-"Morphological" rules: tdl

```
dative-alt-lex-rule := const-ltol-rule &
                       ditrans-verb-lex &
[ SYNSEM.LOCAL [ ARG-S < #subj,
                         [ ...INDEX #i1 ],
                         [ ...INDEX  #i2 ]>,
                 CONT #cont ],
  DTR pptrans-verb-lex &
     [ SYNSEM.LOCAL [ ARG-S < #subj,
                              [...INDEX #i2 ],
                              [INDEX #i1 ]>,
                      CONT #cont ] ] ].
```

"Morphological" rules: tdl

- Ex: Japanese nominative case:

```
nom-case-lex-rule :=
%suffix (* ga)
infl-ltow-rule &
[ SYNSEM.LOCAL.CAT.HEAD.CASE nom,
  DTR.SYNSEM.LOCAL.CAT.HEAD noun ].
```

- Take an input that is [INFLECTED −] and [HEAD noun]
- Add the suffix *-ga*
- Make its CASE value nom.

Morpho-orthography (1/3)

- In the definition of a lexical rule instance (`irules.tdl`), after `:=` and before the supertypes, put a line starting with `%suffix` or `%prefix`.
- `%suffix`/`%prefix` is followed by a list of pairs of quasi-regular expressions.
- Within each pair, the first member matches the input form the second member describes the output form.
- More general cases to the left, more specific cases to the right.

Morpho-orthography (2/3)

- * represents the null character
- `!` calls letter classes (which need to be defined), e.g.,

  `%(letter-set (!v aeiou))`
- Can also record suppletive forms in `irregs.tab`.

Morpho-orthography (3/3)

```
3sg-v_irule :=
%suffix (!s s) (!ss !ssses) (ss sses)
sing-verb.

past-v_irule :=
%suffix (* ed) (!ty !tied) (e ed)
(!t!v!c !t!v!cced) (give gave)
past-verb.

%(letter-set (!c bdfglmnprstz))
%(letter-set (!s abcedfghijklmnopqrtuvwxyz))
%(letter-set (!t bcdfghjklmnpqrstvwxz))
%(letter-set (!v aeiou))
```

For next time

- Does your language have case?
- Does your language have agreement?
- Understand case/agreement systems and collect paradigms
- Does your language mark number distinctions anywhere (e.g., pronouns, human nouns...)
- If no case or agreement (or not much), adjectives and adverbs