An Introduction to Grammar Engineering using HPSG: DAY 2

Goals:

1. Learn about typed feature structures and unification.
2. Expand the grammar to capture agreement constraints.
3. Expand the grammar to provide an analysis of modification.

Exercises:

1. Bring up the LKB and load grammar2:
   (a) Check out a copy of grammar2 by typing `cvs checkout grammar2` in an xterm window. Alternatively, if you are content you completed the previous exercise sucessfully, you can also do `cvs update grammar1` and then continue work on '`grammar1`'.
   (b) Start emacs and in the Emacs window:
   `<Esc> x lkb`
   (c) Load the grammar by selecting Load / Complete grammar in the 'Lkb Top' window, then double-clicking on the directory 'grammar2' and on the file 'script'.
2. Test the grammar by parsing the sentence *The cat gave that dog to those dogs.*
3. Extend the grammar to capture subject-verb agreement, admitting e.g. *The dog barks.* but not **The dogs barks.*. We will introduce constraints on the SPR attribute of lexical entries requiring that the person-number properties match between head and specifier. Rather than using separate features for number and person, we will use types that combine both properties, allowing a more direct encoding of English inflectional morphology.
   (a) Add this small type hierarchy to the file types.tdl, making *pernum* a subtype of *feat-struc*:

```
        pernum
        /    \
   3sing   non-3sing
```

   (b) Also in the file types.tdl, add the feature AGR to the type *pos*, with its value constrained to be of the type *pernum* that you just added.
   (c) In the file lexicon.tdl, add the appropriate constraint to each verb, restricting the AGR value of its SPR.
   (d) Also in the file lexicon.tdl, add the correct AGR value to each noun.
   (e) Save your changes, then reload the grammar, apply the batch test with the file 'agr.items', examine your results, and make any necessary corrections.
4. Extend your analysis to cope with determiner-noun agreement, admitting e.g. *These dogs bark.* but not *These dog barks.*.
   (a) In the file lexicon.tdl, modify each noun's lexical entry by adding the appropriate constraint on the AGR value of its SPR.
   (b) Also in lexicon.tdl, add the correct AGR value to each determiner.
   (c) Check your revised grammar again using the file 'agr.items', and make any necessary corrections.
   (d) Add some additional test examples to this file with varied combinations of mismatch in agreement among determiners, nouns, and verbs. Then run the batch test and examine the results.

5. Extend the grammar to provide an analysis of modification, admitting sentences like *The dog barks near the cat.* We introduce a new head feature MOD and a new syntactic rule for modifiers, and we make use of the notion of underspecification.

   (a) In the file types.tdl, add the feature MOD to the definition of the type *pos*, with its value constrained to be of **list**, the same type as for the attribute SPR.

   (b) Also in types.tdl, assign an appropriate value for MOD to each of the subtypes of *pos*.

   (c) In the file rules.tdl, add a new rule 'head-modifier' rule somewhat similar to the head-specifier rule, but with the modifier daughter constraining the head daughter.

   (d) In the file lexicon.tdl, add a lexical entry for the preposition *near*.

   (e) Save your changes, then test your revised grammar using the file 'mod.items'. Examine the results, and make any necessary corrections.

   (f) If your analysis does not already admit examples like *The dog near the cat barks.*, modify your grammar appropriately.

   (g) If your analysis provides two parses for the sentence *The dog barks near the cat*, modify your grammar to eliminate one of the two parses, then run the batch parse again with the file 'mod.items', and examine the results.

   (h) Add additional sentences to the file 'mod.items', and notice what happens to the number of analyses as you add prepositional phrase modifiers within a single sentence.