**Practical Grammar Engineering using HPSG**
Exercises day 1: part a

**Goals:**

1. Become familiar with the LKB grammar development system.
2. Learn to extend a grammar by adding lexical entries.
3. Expand the grammar to deal with a ditransitive verb.
4. Expand the grammar to add prepositional phrase arguments.

**Exercises:**

1. Bring up the LKB and load grammar1:
   (a) Checkout a copy of grammar1 by `cvs checkout grammar1` in an X-terminal.
   (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 “grammar1” and on the file “script”. Reassuring messages will appear in the “Lkb Top” window, and a window will pop up showing you the type hierarchy for this small grammar.
2. Test the system by parsing the sentence *the cat chased the dog*:
   (a) With the mouse in the “Lkb Top” window, click on the button Parse.
   (b) Click on the menu item Parse input. . . .
   (c) Type in the sentence *the cat chased the dog* replacing the existing contents of the window.
   (d) Click on the button OK.

   The system will parse the sentence and pop up a window containing a little parse tree for the single analysis of this sentence. Click on the parse tree to get a menu which allows you to enlarge it and look at the nodes.
3. Try the simple batch parsing mechanism:
   (a) In the “Lkb Top” window, click on the button Parse and then on the menu item Batch parse. . . , which will pop up a window asking you for the file to be processed.
   (b) Click on the file “test.items” in your grammar directory. then hit the button OK. This will pop up a new window asking you for the name of the file where the results of the batch run will be stored.
   (c) Enter the name “test.results” and hit the OK button. The system will print the message `Parsing test file` in the `*common-lisp*` emacs buffer when it starts, and will print the message `Finished test file` when it is done.
   (d) Open the file “test.results” from emacs and inspect the parsing results.
4. Add a lexical entry for another animal noun of your choice:
   (a) In the Emacs window, open the file “lexicon.tdl” for editing (see the emacs handout)
   (b) In the lexicon.tdl buffer that you get, copy the five lines that define the lexical entry for **dog** and modify your copy to make the value of ORTH appropriate for another animal.
   (c) Save the changed version of the file.
5. Reload the grammar and test the effect of your addition:

   (a) In the “Lkb Top” window, select Load / Reload grammar.
   (b) Parse the sentence *the cat chased the ⟨your-animal⟩*
   (c) Add this sentence to the test.items file and rerun the batch check.

6. Investigate the grammar in order to get an intuitive idea of how it works (formal details will be discussed later). In particular, look at the following sentences and try and decide why they do or do not parse:

   - the cat barks
   - the cat chased
   - cat barks
   - the cat bark
   - bark

   Note that the parse chart will be available even if the sentence didn’t parse, and you can click on the nodes in that to display feature structures. Notice that the grammar is parsing some sentences incorrectly (overgeneration) and failing to parse some sentences that should parse (undergeneration).

7. The rule that is needed for ditransitives is in the grammar, but there are no lexical entries that utilize it. Add an entry for *gave* which takes two noun phrase complements (i.e., the entry you would need to parse *that dog gave the cat the ⟨your-animal⟩*).

   (a) Copy the entry for *chased* in lexicon.tdl
   (b) Replace the orthography value as before.
   (c) Add an extra element to the COMPS list, which will be a duplicate of the one that is already there. Note: lists are delimited by `<` and `>` and that elements on lists are separated by commas.
   (d) Test by parsing *that dog gave the cat the ⟨your-animal⟩*. Also test for overgeneration by confirming that you cannot parse *that dog gave the cat.*
   (e) Add appropriate sentences to test.items

8. Add a new type and two new lexical entries in order to parse *that dog gave the cat to the ⟨your-animal⟩*

   (a) Add the type **prep** as a new subtype of the type **pos** to the file types.tdl, by copying the type description for **noun** and replacing **noun** with **prep**.
   (b) Add a lexical entry for the preposition *to.* This should be similar to the entry for *chased* in that *to* will take a single noun phrase complement, but the value for HEAD should be **prep** and the value of SPR should be the empty list (i.e., `<>`).
   (c) Add another lexical entry for *gave.* You can copy your existing entry for *gave* but you will need to use a different identifier (i.e., the thing to the left of the `:=`), e.g., `gave_2`. You also need to change the second element in COMPS to make this entry require a PP (we won’t bother about making sure it’s a *to*-PP yet).
   (d) Add several test items, both grammatical and ungrammatical, to the file “test.items” which will allow you to check the correctness of your additions to the grammar.
   (e) Run the batch parsing utility again on the file “test.items”, and examine the results.
   (f) Celebrate as appropriate.