1 Background

Like many languages, English has constructions where at least one of the syntactic arguments of a predicate does not show up right beside its predicate, but somewhere farther away. One class of these constructions is called long-distance dependencies, illustrated by the following examples:

(1) *That cat, the dog wanted me to give to the aardvark.*
(2) *Examples like this, I’m sure nobody ever really says.*

Long-distance dependencies are discussed in more detail in Chapter 14 of (Sag, Wasow, & Bender, 2003); for the current exercise, it will be enough to remember that a sentence-initial phrase like *that cat* can be analyzed as providing the *filler* for a *gap* (a missing argument) somewhere later in the sentence.

2 The Starting Grammar

You can use the grammar you started with yesterday (grammar06), but do
```
$ cd grammar06; cvs update
```
Or you can get a fresh copy of that grammar by renaming yours, and then once again doing the following:
```
$ cvs -d :ext:[username]@login.coli.uni-saarland.de:/proj/delphin/CVS co ge-ss10/grammar06
```

3 Implementation

- Examine the phrase types that the Matrix makes available for these unbounded dependencies, as discussed in the lecture. Notice that the information about a missing complement is introduced by a unary rule that ‘discharges’ the complement and identifies some of those properties as a non-empty value in the *SLASH* attribute.

- Add an English-specific rule that makes use of these types, to enable the extraction of a complement, so your grammar can analyze *chased* as a verb phrase with a non-empty *SLASH*.

- Also add the English-specific rule that combines the *filler* phrase and a sentence with a gap, so your grammar can combine a noun phrase like *that cat* with the phrase *the dogs chased*. Notice that the phrase *the dogs chased* is constructed using the ordinary subj-head rule, which inherits from the type head-valence-phrase, ensuring that the SLASH value is passed up the tree.

- Test your changes on the items in the file ‘gap.items’, and tune your implementation accordingly.

As usual, document your analysis for each step.