Introduction to CL I (684.01) OSU Linguistics
Detmar Meurers Winter 2007

Exercise sheet 3

(Submit as a plain text email message to dm@ling.osu.edu before or on Monday, Feb. 5)

1. We saw in class that the standard DCG implementation produces translations in a way exemplified by the following:

   ```
   s --> np, vp
   s(S0,S) :- np(S0,S1), vp(S1,S).
   ```

   and a DCG is used as a recogniser by giving goals such as:

   ```
   ?- s([john,smiled],[]).
   ```

   Consider a proposed alternative implementation that instead produced a translation of one of the following kinds:

   ```
   s(S0,S) :- np(S1,S), vp(S0,S1).
   s(S0,S) :- vp(S0,S1), np(S1,S).
   s(S0,S) :- vp(S1,S), np(S0,S1).
   ```

   Assuming that the same kind of query is to be presented,

   (a) Which of these are correct translations, given the **meaning** of the original phrase structure rule?

   (b) What sort of **recognition behaviour** do they produce (i.e., in what order does recognition take place)?

2. It is cumbersome to distinguish verbs according to their subcategorization requirements and repeat this distinction in the rules realizing the head and its arguments, as illustrated by the following grammar:

   ```
   vp --> v(intrans), [].
   vp --> v(trans), np.
   vp --> v(ditrans), np, np.

   v(trans) --> [saw].
   v(intrans) --> [left].
   v(ditrans) --> [gave].
   np --> [john]; [mary]; [froggy].
   ```

   Write and test a strongly-equivalent grammar using meta-variables in a DCG to get over this duplication. Provide **your test grammar** and a **short explanation** of why it works.

3. Write and test a DCG which will recognize all sentences in the first column and reject all those in the second:

Mary has laughed.	*Mary has laughing.*
Mary is laughing.	*Mary are laughing.*
Paul is a duck.	*Paul is a ducks.*
We see three ducks leave.	*Paul is three ducks.*
We see her leave.	*We see her leaves.*

   and it should provide two distinct analyses for *Paul saw her duck.*

   Note that if multiple solutions do not require the arguments of the queried predicate to be bound different, prolog only returns a single `true`; please verify that your grammar does return two distinct analyses for the above sentence by tracing it with the SWI debugger.