Basic DCG notation for encoding CFGs

A DCG rule has the form "*LHS* `-->` *RHS*." with

- *LHS*: a Prolog atom encoding a non-terminal, and
- *RHS*: a comma separated sequence of
  - Prolog atoms encoding non-terminals
  - Prolog lists encoding terminals

Examples for some context free grammar rules:

- S $\rightarrow$ NP VP
  `s --> np, vp.`
- S $\rightarrow$ NP thinks S
  `s --> np, [thinks], s.`
- S $\rightarrow$ NP picks up NP
  `s --> np, [picks, up], np.`
- S $\rightarrow$ NP picks NP up
  `s --> np, [picks], np, [up].`
- NP $\rightarrow$ ϵ
  `np --> [].`

More complex terms in DCGs

Non-terminals can be any Prolog term, e.g.:

```
s --> np(Per,Num),
      vp(Per,Num).

s(s_node(NP,VP)) --> np(NP),
                     vp(VP).
```

Restriction:

- The *LHS* has to be a non-variable, single term (plus possibly a sequence of terminals).

Additional notation for the RHS of DCGs

The *RHS* can include

- **disjunctions** expressed by the ";" operator, e.g.:

  ```
  vp --> vintr;
         vtrans, np.
  ```

- **groupings** are expressed using parenthesis "( )"
- **extra conditions** in the form of prolog relation calls enclosed in "{ }", e.g.:

  ```
  s --> np(Case), vp,
        {check_case(Case)}.

  s --> {write('rule 1'), nl},
        np,
        {write('after np'), nl},
        vp,
        {write('after vp'), nl}.
  ```

- the **cut** "!" (can occur without enclosing "{}").

Meta-variables

On the *RHS*, variables can be used for non-terminals and terminals, i.e. as meta-variables. E.g.:

```
verb([up]) --> [pick].

vp --> verb(Particle),          % pick
       np,                      % the ball
       Particle.                % up
```

Restriction:

- The value of the variable has to be known at the time Prolog attempts to prove the subgoal represented by the variable.

Towards a basic DCG for English
X-bar Theory

Generalizing over possible phrase structure rules, one can attempt to specify DCG rules fitting the following general pattern:

$\mathsf{X}^2 \rightarrow$ specifier^2 X^1
$\mathsf{X}^1 \rightarrow \mathsf{X}^1$ modifier^2
$\mathsf{X}^1 \rightarrow$ modifier^2 X^1
$\mathsf{X}^1 \rightarrow \mathsf{X}^0$ complement^{2*}

To turn this general X-bar pattern into actual DCG rules,

- X has to be replaced by one of the atoms encoding syntactic categories, and
- the bar-level needs to be encoded as an argument of each predicate encoding a syntactic category.

Noun, preposition, and adjective phrases
Some example rules

```
n(2,Num) --> pronoun(Num).
n(2,Num) --> proper_noun(Num).
n(2,Num) --> det(Num), n(1,Num).
n(2,plur) --> n(1,plur).
n(1,Num) --> pre_mod, n(1,Num).
n(1,Num) --> n(1,Num), post_mod.
n(1,Num) --> n(0,Num).
...

p(2,Pform) --> p(1,Pform).
p(1,Pform) --> adv, p(1,Pform).
               % slowly past the window
p(1,Pform) --> p(0,Pform), n(2,_).
...

a(2) --> deg, a(1). % very simple
a(1) --> adv, a(1). % commonly used
a(1) --> a(0).
```

Verb phrases and sentences

Some example rules

```
v(2,Vform,Num) --> v(1,Vform,Num).
v(1,Vform,Num) --> adv,
                   v(1,Vform,Num).
v(1,Vform,Num) --> v(1,Vform,Num),
                   verb_postmods.
v(1,Vform,Num) --> v(0,intrans,Vform,Num).
v(1,Vform,Num) --> v(0,trans,Vform,Num),
                   n(2).
v(1,Vform,Num) --> v(0,ditrans,Vform,Num),
                   n(2),
                   n(2).
...

s(Vform) --> n(2,Num),
             v(2,Vform,Num).
```