DCG's

Recall

```
Sentence(Input, AfterS) :-    noun_phrase(Input, AfterN),
                              verb_phrase(AfterN, AfterS).
```

in DCG syntax, we could have written

```
Sentence --> noun_phrase, verb_phrase
```

We can also use:

```
verb_phrase --> verb | noun_phrase
```

If we have a terminal we can use

```
verb --> [eat] | [sleep]
```

translate

```
verb([eat | afterV], AfterV).
verb([sleep] | AfterV], AfterV).
```

Aside Prolog Strings

```
|?- x = "hi".
yes     x = [104, 105]
```

so

verb --> "eat" + verb([[102, 98, 115] | AfterV], AfterV)

<u>What's new in DCG's over Context Free Grammars</u>

can also have rules containing untranslated prolog goals.

Example: Make a translator from arabic numbers -> english.

55 fifty-five

```
num(0) --> [zero].
```

gets translated to num(0, [zero | AfterNum], AfterNum)

```
num(N) --> xx(N).

xx(N) --> digit (N) | teen(B) | tens(T), digit(N), {N is T + N1}.
digit(0) --> [0].
digit(1) --> [one].
digit(2) --> [two].
…
…
digit(9) --> [nine].
teens(10) --> [ten].
teens(11) --> [eleven].
…
…
teens(19) --> [nineteen].
tens(20) --> [twenty].
…
tens(90) --> [ninety].
```

|? – num(27, X, [ ]).
yes x = [twenty, seven]

Knowledge Representation

Need common techniques / strategies for representing things about the world in 1st order logic.

Situation calculus – Used in planning
- Has 3 main components
  - Situations – Logical term/constant used to express state.
  - Example:
    - Might have a constant s0 to denote initial state of the system
  - Example:
    - Might have a term result(Action, State) which returns the state that results from some action.
  - Fluents – Functions or predicates that map from one situation to another in this calculus.
  - Example:
    - Result from above as a function is a fluent
  - Example:
    - (A predicate Fluent) not (holding (gold, S)).
  - A temporal/Eternal predicate – predicates whose value are independent of the situation.
    - is_gold(gold). always true since gold is always gold regardless of the situation.

What kinds of things do we want to do with situation calculus?
1. Predict all of the things that would follow if we did a given sequence of actions (called projection)
2. Compute a sequence of actions that achieve a desired set of consequences. (planning)

<u>How to create toy worlds using the situation calculus.</u>
To do this, we need to choose axioms for our world.
Typical kinds of axioms
1. Possibility
   preconditions => possible(a, s)
   <u>Effect Axioms</u>
   possible(a, s) => what would be the result of taking action a in state s
2. to be continued…