Natural Language with Prolog

Agenda

- Summarize Situation Calculus/planning
- Discuss Assignment 3
- Discuss Lab 6
- Lesson: Natural Language with Prolog

Summary of Chatbot operation

1. User enters a list of words
2. List of words is parsed to achieve the logical form based on semantics
3. Logical form is converted to Clausal Form
4. Clausal Form is asserted to prolog database (assertions) or issued as a query (questions)

talk predicate

```
talk(Sentence, Reply) :-
    % parse the sentence
    parse(Sentence, LF, Type),
    % convert the FOL logical form into a Horn
    % clause, if possible
    clausify(LF, Clause, FreeVars), !,
    % concoct a reply, based on the clause and
    % whether sentence was a query or assertion
    reply(Type, FreeVars, Clause, Reply).
```

Intermediate steps

If we want to see the intermediate steps of the talk predicate, we can use a query like this (an assertion):

?- parse([a, man, observed, a, particle],LF,Type),clausify(LF,Clause,Freevars),reply(Type, FreeVars, Clause, Reply).

LF = exists(man1, @man(man1)&exists(particle1, @particle(particle1)& @observe(man1, particle1))),

Type = assertion,

Clause = (man(man1), particle(particle1), observe(man1, particle1)),

Freevars = [],

Reply = asserted((man(man1), particle(particle1), observe(man1, particle1))) .

Intermediate steps (cont'd)

Or like this (question):

?- parse([who,observed,particle1],LF,Type),clausify(LF,Clause,Freevars),reply(Type, FreeVars, Clause, Reply).

LF = @observe(_A, particle1)=> @answer(_A),

Type = query,

Clause = (answer(_A):-observe(_A, particle1)),

Freevars = [],

Reply = answer([man1]) .

Clausify

The clausify predicate translates logical form into the equivalent clausal form.

Recall that prolog can deal with only Horn clauses.

Example of statement that cannot be converted to a Horn clause:

Canada is a country or the sky is blue

Logical form: `country(canada) or blue(sky)`

Clausal form (prolog syntax): `country(canada);blue(sky).`

We cannot say this in prolog because it is not a horn clause.

Clausal Normal Form (clausify)

Situation Calculus formulae can be converted to clausal form:

1. Eliminate implications
2. Move negation inwards
3. Standardize variables apart
4. Skolemize existentials
5. Move universal quantifiers outwards
6. Distribute "and" ∧ over "or" ∨

CNF Step 4: Skolemize

Replace existentially quantified variables with skolem functions

Each universally quantified variable whose scope includes the existential becomes a parameter of the skolem function

The idea is that if we know something exists, we can give it a name (any name that works for us is fine)

$\forall x \exists y P(x, y)$ becomes $\forall x P(x, a(x))$ where new name a depends on x

$\exists y \forall x P(x, y)$ becomes $\forall x P(x, a)$ new name a doesn't depend on x

Examples:

$\forall x \exists y\ mother(x, y)$ becomes $\forall x\ mother(x, mom(x))$

$\exists x\ moon(x)$ becomes moon(the_moon)

Prolog code

```
% asserts that F is clausal form of exists(X,F0)
clausify(exists(X,F0),F,V) :- % something like exists(X,man(X)& _)
    F0 = @G&_,     % G is something like man(X)
    skolem(G,X),
    clausify(F0,F,V).  % F0 is now @man(man2)&_ and yellow part gone
skolem(G,X):-
    functor(G,Name,1),     % functor(man(X),man,1), see functor/3 doc
    gensym(Name,X),        % X becomes manN for some integer N
    assert(pn(X,X)).       % assert(pn(man2,man2)) when N is 2
```