Prologging Languages

Hannah Twigg-Smith and Philip Seger

What is Prolog?

- Declarative logic language
- Expressed through relations as **facts** and **rules**
- Widely used for artificial intelligence and relation logic in databases
- Has lots of flavors (such as Datalog)

Data types

- **Atom** - general term with no meaning: `mia`, `'test'`, `'other things'`
- **Numbers** - floats and ints
- **Variables** - anything that starts with a capital or underscore: `Test`, `_moreTest`
- **Facts**
  - Facts start with a lowercase letter and end with a period: `canSwim.` or `isFlying.`
  - Facts can contain arguments - which can be atoms and numbers;
  - `canSwim(mia).` `isMother(becky, jon).` `otherExample(isthis).`

Structure

- Started off with a Homework 3-esque structure with a shell, parser, expression classes, and value classes
- Had problems with storing values in the environment
  - Instead of spending time figuring this out, we reverted to a dictionary with nested lists of strings
- The bulk of the code is the main unification function that evaluates each query by either verifying a relation exists or finding potential relations in the environment

Environment

- Currently, the environment takes the form of a dictionary
- Keys are relations
- Values are the relation arguments
- Example:

```
env = {
    “mother” : [[“mia”, “jon”],[“mia, “mary”]],
    “person” : [[“pip”], [“hannah”]]
}
```

Rules

fact:

```
> person(hannah).
```

rule:

```
> human(A) :- person(A).
```

check (for our language):

```
> human(hannah)?

yes
```

Adding complexity

```
> inProlang(hannah).

> inProlang(philip).

> partners(hannah, philip).

> doingProlang(X,Y) :- inProlang(X),inProlang(Y).

> partnerProlang(X,Y) :- doingProlang(X,Y),partners(X,Y).

> partnerProlang(hannah, philip)?
```

Future work

Done:

- Implemented rules, facts, and queries
- Evaluation of facts with variables (isWorking(X)?)

Possible additions:

- Make variables work with rules
- Arrays and numbers