Assignment 2: A Fiasco playset in Prolog

Out: Wednesday, April 15, 2015
Due: Wednesday, April 22, 11:59pm

Overview

Note: if you missed class today, read over the slides for "Unit 6 – PCG" before doing this assignment.

In this assignment, you and your teammates will create a Prolog formalization of a *Fiasco* playset suitable for the automatic setup generator. You should work in teams of 3 or 4. You can work in smaller teams if you want, but you can't really play *Fiasco* with 2 people and if try to play it with more than 4, it will take a long time.

Getting started

You should start by reading examples of *Fiasco* playsets at:

http://www.bullypulpitgames.com/wiki/index.php?title=Fiasco_Playsets

The game itself is available from Amazon, DriveThruRPG, and locally at Dice Dojo. However, you do not need to buy the game for the class (although you have to read *some* storytelling game, be it *Fiasco* or something else).

What you need to write

Any Fiasco playset defines a set of:

- Relationships
  You can declare these as follows:
  - relation(NameOfRelation)
    Says that NameOfRelation is a valid kind of relationship between characters in this playset.
  - symmetric(NameOfRelation)
    Says that NameOfRelation is a valid kind of relationship between characters in this playset and that it is symmetric, meaning that if it holds between characters X and Y, it also hold between Y and X. Siblings is a symmetric relation, parent_of is not.
  - roles_relation(Role1/Role2)
    Says that one kind of relationship between characters is for one of the characters to take on the role Role1 and the other to take on the role Role2. For example boss/employee, or waitron/customer.

- Needs
  You can declare these by just saying need(NameOfNeed).
- Objects
  You can declare these by just saying object(NameOfObject).
- Locations
  You can declare these by just saying location(NameOfLocation).

However, since we're trying to automate the generation of setups, we also need to provide information about what kinds of relationships and so on are incompatible with one another. You specify these using the **implies** and **contradiction** predicates in Prolog:

- contradiction(Fact1, Fact2)
  Says that we can't have a setup where both facts are true. For example:
  ```
  contradiction(relationship(X, parent_of, Y),
                relationship(Y, parent_of, X)).
  ```
- implies(Fact1, Fact2)
  Says that if Fact1 is true in the setup, then Fact2 is also true in it. For example:
  ```
  implies(relationship(X, parent_of, Y),
          adult(X)).
  ```

That's basically what you need to specify. There are a few extra things you can specify that act as shorthands for implies and contradiction:

- generalizes(Relation1, Relation2)
  Says that Relation1 holds between two characters, then Relation2. So saying:
  ```
  generalizes(sibling, family).
  generalizes(parent_of, family).
  ```
  Saves you having to type:
  ```
  implies(relationship(X, sibling, Y),
          relationship(Y, family, X)).
  implies(relationship(X, parent_of, Y),
          relationship(Y, family, X)).
  ```
- conflicting_roles(X, Y)
  Says that a character can't have both roles at the same time. So if we say conflicting_roles(employee, king), then a character can't have the boss/employee relationship with one character, while also having the king/subject relation with another.

That's it. Have fun.

Grading

You will not be graded on whether your playset is good in the sense of being funny or dramatic, or fun to play, although since you'll be playing it, it would be good to try to make it be fun. For purposes of grading, you just need to have enough stuff in

your playset, and the amount of stuff depends on the number of people in your team. In general, each person in your team should make:

- 5 kinds of relationships
- 5 needs
- 5 objects
- 5 locations

That much is really easy. However, there should also be restrictions on what combinations are possible so as to prevent nonsensical combinations. For purposes of grading, I expect to be able to run your playset a few times and not see any glaring contradictions. If I do, I'll deduct points. And in any case, I would expect to see at least 5 contradiction or conflicting_roles declarations per teammate and at least 5 implies declarations per teammate.

Running your playset

Start up SWI Prolog and load in Solver.pl and then your playset file. That is, either open the file in the editor and do compile buffer on each file, or do consult('filename'), if you aren't using the SWI Prolog IDE. It's important to load Solver.pl first, since it includes declarations that need to be loaded before the other file.

You can experiment with the playset that's included with the assignment if you like.

To test your playset, just run the setup predicate, passing it the names of the characters as arguments. For example:

```
?- setup(ken, larry, ian).
```

It will randomly choose a setup based on your playset and print it, along with all the inferences it made along the way. Be sure to try your playset many times, looking for nonsensical setups that it generates. When you find one, add some implication or contradiction rules to prevent it from generating them in the future, and then either rerun Compile Buffer, if you're using the SWI editor, or reload it using reconsult, if not.

Turning it in

When your team is happy with its playset, upload it to Canvas.