Solving murder with Prolog

Dec 21, 2018

In our company we usually have a fun quiz every sunday and this was one of them

The puzzle

To discover who killed Mr. Boddy, you need to learn where each person was, and what weapon was in the room. Clues are scattered throughout the quiz (you cannot solve question 1 until all 10 are read).

- To begin, you need to know the suspects. There are three men (George, John, Robert) and three women (Barbara, Christine, Yolanda). Each person was in a different room (Bathroom, Dining Room, Kitchen, Living Room, Pantry, Study). A suspected weapon was found in each room (Bag, Firearm, Gas, Knife, Poison, Rope). Who was found in the kitchen?
- Clue 1: The man in the kitchen was not found with the rope, knife, or bag. Which weapon, then, which was not the firearm, was found in the kitchen?
- Clue 2: Barbara was either in the study or the bathroom; Yolanda was in the other. Which room was Barbara found in?
- Clue 3: The person with the bag, who was not Barbara nor George, was not in the bathroom nor the dining room. Who had the bag in the room with them?
- Clue 4: The woman with the rope was found in the study. Who had the rope?
- Clue 5: The weapon in the living room was found with either John or George. What weapon was in the living room?
- Clue 6: The knife was not in the dining room. So where was the knife?
- Clue 7: Yolanda was not with the weapon found in the study nor the pantry. What weapon was found with Yolanda?
- Clue 8: The firearm was in the room with George. In which room was the firearm found?
- It was discovered that Mr. Boddy was gassed in the pantry. The suspect found in that room was the murderer. Who, then, do you point the finger towards?

I suck at these kind of puzzles (actually most of the puzzles), and those can take hours and hours of thinking, but there's always Prolog to the rescue! Prolog can help solving these kinds of reasoning puzzles and we will see how.

prolog 101

Install SWI-Prolog

```
~> swipl
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 6.6.6)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

?- write('Hello, World!').
Hello, World!
true.
?- write('Hello,'), nl, write('world').
Hello,
world
true.
?- X is 3*4 + 2.
X = 14.
```

- `swipl` is our prolog interpreter binary
- `write` is called a `functor` and represented as `write/1` means it takes 1 argument. (same concept in erlang or elixir to add the number of arguments to the function name)
- `nl` used to print newline
- to execute sequence of commands you use `,` which is the AND operator also if it fails the whole computation fails
- `is` assginment operator followed by math expression
- variables are Capitalized `X` not x

Knowledge base

Prolog is all about stating facts, composing facts and querying them.

create a file `hello.pl`

```
friend(john, julia).
friend(john, jack).
friend(julia, sam).
friend(julia, molly).

loves(john, julia).
loves(julia, sam).
loves(sam, julia).

male(brad).
male(john).
male(jim).
male(alfred).
female(marry).
child(brad, alfred).
child(john, jim).
child(john, marry).
```

- to load it we use `[hello].` : notice the . in the end
- use `listing` to list all the facts in the knowledge base

```
?- [hello].
% hello compiled 0.00 sec, 3 clauses
true.

?- listing(friend).
friend(john, julia).
friend(john, jack).
friend(julia, sam).
friend(julia, molly).

true.

?- listing(loves).
loves(john, julia).
loves(julia, sam).
loves(sam, julia).

true.
```

Querying the facts

After stating the facts in our knowledge base we can go ahead and ask prolog about the truth or what it can deduce from the facts we gave it.

```
?- friend(john, julia).
true .

?- friend(john, jack).
true.

?- loves(john, julia).
true.

?- loves(john, sam).
false.
```

We can go for more complex queries like asking (Who is friend with john or Who loves julia)

```
?- friend(john, Who).
Who = julia ;
Who = jack.
```

```
?- listing(child).
child(brad, alfred).
child(john, jim).
child(john, mary).

true.

?- child(john, X).
X = jim ;
X = mary.
```

Is John friendzoned?

We defined `friend` relation `friend(john, julia)` It says `john is friend with julia`, but for prolog that doesn't mean `julia is friend with john` you need to add another fact saying `friend(julia, john)`, also we already defined the `child` relations, for sure we don't want to duplicate it again and flip the arguments to define `parent` relations. We don't want to write

```
child(brad, alfred).
child(john, jim).
```

```
child(john, mary).

parent(alfred, brad).
parent(jim, john).
parent(mary, john).
```

Prolog helps by allowing inference rules.

`rule :- stmt1, stmt2,...`

rule is true if all of the inner statements are true (ANDed together with `,`)

```
friend(X, Y) :- friend(Y,X).
parent(X, Y) :- child(Y,X).
father(X, Y) :- child(Y,X), male(X).
mother(X, Y) :- child(Y,X), female(X).
friendzoned(X) :- loves(X, Y), \+ loves(Y,X).
```

- `friend(X,Y)` is a rule that is true if also `friend(Y,X)`)
- `parent(X,Y)` is true if `child(Y,X)` is defined
- `father(X,Y)` is true if `parent(X,Y)` and `male(X)` are defined
- `mother(X,Y)` is true if `parent(X,Y)` and `female(X)` are defined.
- `friendzoned(X)` is true if X loves `SOMEONE Y` and that Y doesn't love X (notice the hidden variable Y?)

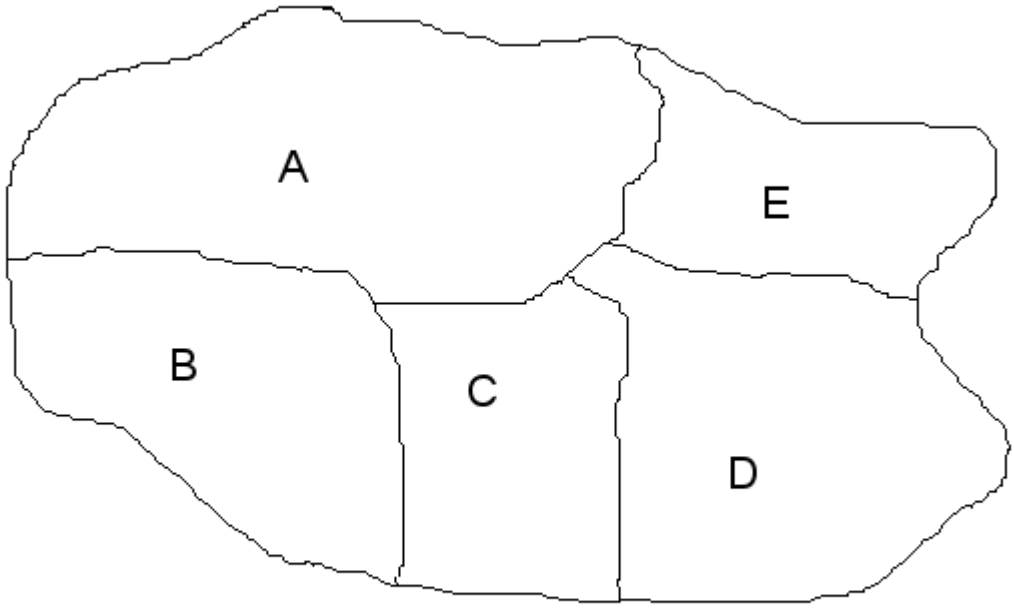
```
?- friend(julia, john).
true .
?- male(jim).
true.

?- parent(jim,X).
X = john.

?- father(jim, X).
X = john.

?- mother(X, john).
X = marry.

?- mother(marry,X).
X = john.

?- mother(marry, john).
```

```
true.

?- loves(julia, X).
X = sam.

?- friendzoned(julia).
false.

?- friendzoned(john).
true.
```

OK now we know enough prolog to be dangerous. we will warm up with map coloring problem

Map coloring

First we will start with solving a map coloring problem. A famous problem in mathematics concerns coloring regions on the maps. it is required that any two adjacent regions may not have the same color.



So our reasoning should be, we have 1- Variables: areas we want to color A,B,C,D,E 2- Domain: the range of values that can be assigned to our variables) and that would be (red, blue, green) 3- Stating the constraints that no adjacent areas can have the same color.

Domain

Let's define the domain of our Areas (red, green, blue)

```
color(red).
color(green).
color(blue).
```

Just like that

Asking for solution

```
colorify(A,B,C,D,E) :-

    color(A), color(B), color(C), color(D), color(E),
    \+ A=B, \+ A=C, \+ A=D, \+ A=E,
    \+ B=C, \+ C=D, \+ D=E.
```

Here we define our solution as a rule `colorify` that has 5 `variables` A,B,C,D,E and in the body we assign the domain color `red, blue, green` to our variables `A,B,C,D,E` and state the constraints that A not equal to B an A not equal to C, ... etc

> `\+ X=Y` *means X is not equal to Y*

Prolog now will keep generating values like (red, blue, green) and assigns them our variables `A,B,C,D,E` until our constraints are met

```
?- [mapcoloring]
|    .
true.

?- colorify(A,B,C,D,E)
|    .
A = red,
B = D, D = green,
C = E, E = blue ;
A = red,
B = D, D = blue,
C = E, E = green ;
A = green,
B = D, D = red,
```

```
C = E, E = blue ;
A = green,
B = D, D = blue,
C = E, E = red ;
A = blue,
B = D, D = red,
C = E, E = green ;
A = blue,
B = D, D = green,
C = E, E = red
```

```
color(red).
color(green).
color(blue).

colorify(A,B,C,D,E) :-
    color(A), color(B), color(C), color(D), color(E),
    \+ A=B, \+ A=C, \+ A=D, \+ A=E,
    \+ B=C, \+ C=D, \+ D=E.
```

but we aren't here to solve map coloring.. let's get back to the murder.

The murder

To begin, you need to know the suspects. There are three men (George, John, Robert) and three women (Barbara, Christine, Yolanda). Each person was in a different room (Bathroom, Dining Room, Kitchen, Living Room, Pantry, Study). A suspected weapon was found in each room (Bag, Firearm, Gas, Knife, Poison, Rope).

Who was found in the kitchen?

Domain

From that we can infer that our domains contains `man`, `woman` `person` or suspect, `location` and `weapons` and our variables are (A,B,C,D,E,F) need to represent (a Person and a location and a weapon) with some constraints that will be revealed in the upcoming clues

```
man(george). man(john). man(robert).
woman(barbara). woman(christine). woman(yolanda).
person(X):- man(X).
```

```
person(X):- woman(X).
location(bathroom). location(dining). location(kitchen). location(livingroom). lo
weapon(bag). weapon(firearm). weapon(gas). weapon(knife). weapon(poison). weapon(
```

`uniq_ppl` rule generates unique values for our variables (A,B,C,D,E) such that they are all unique.

```
uniq_ppl(A,B,C,D,E,F):- person(A), person(B), person(C), person(D), person(E), pe
```

Solution

We start by defining `murderer` rule with unique people `in locations` and unique people `having weapons` and now will specify the relation between the people in the locations with those having weapons

> *Note we are still working against 6 suspects.*

Entry

```
murderer(X) :-
   uniq_ppl(Bathroom, Dining, Kitchen, Livingroom, Pantry, Study),
   uniq_ppl(Bag, Firearm, Gas, Knife, Poison, Rope),
```

To easily reason about the variables like `Bathroom, Dining, Firearm, Gas` we say

- Bathroom: the suspect (man or woman) in Bathroom
- Firearm: the suspect (man or woman) has a Firearm .. etc, you can also think of it as a grid

now we will keep adding constraints `after the last comma in our murderer rule`

Clue 1

The man in the kitchen was not found with the rope, knife, or bag. Which weapon, then, which was not the firearm, was found in the kitchen?

```
% 2. Clue 1: The man in the kitchen was not found with the rope, knife, or bag.
% Which weapon, then, which was not the firearm, was found in the kitchen?

   man(Kitchen),
   \+Kitchen=Rope, \+Kitchen=Knife, \+Kitchen=Bag, \+Kitchen=Firearm,
```

So we say the one the `Kitchen` is variable satisfying `man` fact (defined in our domain) and we state that whoever `man` in the `Kitchen` doesn't have any of (`Rope` , `Knife` , `Bag` , `Firearm`)

Clue 2

Clue 2: Barbara was either in the study or the bathroom; Yolanda was in the other. Which room was Barbara found in?

So we can say it was a `woman` in the `Study` and a `woman` in the `Bathroom` AND it wasn't `christine` and we cross the other options for `Barbara` (Kitchen, Dining, Livingroom, Pantry)

```
% % 3. Clue 2: Barbara was either in the study or the bathroom; Yolanda was in th
% % Which room was Barbara found in?
    woman(Bathroom), woman(Study), \+christine=Bathroom, \+christine=Study,
    \+barbara=Dining, \+barbara=Kitchen, \+barbara=Livingroom, \+barbara=Pantry,
```

Clue 3

Clue 3: The person with the bag, who was not barbara nor George, was not in the bathroom nor the dining room. % % Who had the bag in the room with them?

```
% % 4. Clue 3: The person with the bag, who was not Barbara nor George, was not i
% % Who had the bag in the room with them?

    \+barbara=Bag, \+george=Bag, \+Bag=Bathroom, \+Bag=Dining,
```

- `Barbara` not the one with bag
- `george` isn't the one with the bag
- the one with the `Bag` isn't the one in the Bathroom AND isn't the one in the `Dining`

> *Edit: thanks to the nice people on reddit corrected the misinterpretation I had for clue no.3*

Clue 4

Clue 4: The woman with the rope was found in the study.Who had the rope?

- The one with the `Rope` is `woman`
- She was found in the `Study`

```
% % 5. Clue 4: The woman with the rope was found in the study.
% % Who had the rope?

  woman(Rope), Rope=Study,
```

Clue 5

Clue 5: The weapon in the living room was found with either John or George. What weapon was in the living room?

- `man` in `Livingroom`
- `Livingroom` isn't `robert`

```
% % 6. Clue 5: The weapon in the living room was found with either John or George
% % What weapon was in the living room?
    man(Livingroom), \+Livingroom=robert,
```

Clue 6

Clue 6: The knife was not in the dining room.So where was the knife?

```
% % 7. Clue 6: The knife was not in the dining room.
% % So where was the knife?
    \+Knife=Dining,
```

- the suspect with `Knife` wasn't in `Dining`

Clue 7

Clue 7: Yolanda was not with the weapon found in the study nor the pantry. What weapon was found with Yolanda?

```
% % 8. Clue 7: Yolanda was not with the weapon found in the study nor the pantry.
% % What weapon was found with Yolanda?
    \+yolanda=Pantry, \+yolanda=Study,
```

- `yolanda` isn't the one in `Pantry`
- `yolanda` isn't the one in `Study`

Clue 8

```
% % 9. Clue 8: The firearm was in the room with George.
% % In which room was the firearm found?
    Firearm=george,
```

- `george` is the perrson with the `Firearm`

Clue 9

```
% % 10. It was discovered that Mr. Boddy was gassed in the pantry. The suspect fo
% % Who, then, do you point the finger towards?
Pantry=Gas, Pantry=X, write("KILLER IS :"), write(X), nl, writeanswers(Bathroom,
```

- `Gas` in `Pantry` so those are equal
- `Pantry` has the murderer X
- we write the murderer X name using `write`
- we write the the variables `Bathroom, Dining, Kitchen, Livingroom, Pantry, Study, Bag, Firearm, Gas, Knife, Poison, Rope` using `writeanswers` which is defined like so

```
writeanswers(Bathroom, Dining, Kitchen, Livingroom, Pantry, Study, Bag, Firea
  write("Bathroom: "), write(Bathroom), nl,
  write("Dining: "), write(Dining), nl,
  write("Livingroom: "), write(Livingroom), nl,
  write("Pantry: "), write(Pantry), nl,
  write("Study: "), write(Study), nl,
  write("Kitchen: "), write(Kitchen), nl,

  write("Knife: "), write(Knife), nl,
  write("Gas: "), write(Gas), nl,
  write("Rope: "), write(Rope), nl,
  write("Bag: "), write(Bag), nl,
  write("Poison: "), write(Poison), nl,
  write("Firearm: "), write(Firearm), nl.
```

Who is the murderer?

```
 ?- [crime2].
true.
?- murderer(X).
KILLER IS :christine
Bathroom: yolanda
```

```
Dining: george
Livingroom: john
Pantry: christine
Study: barbara
Kitchen: robert
Knife: yolanda
Gas: christine
Rope: barbara
Bag: john
Poison: robert
Firearm: george
X = christine ;
```

Code is available here probably can be much better as I'm no expert in prolog :)

---

xmonader

xmonader
xmonader@gmail.com

xmonader
xmonader

This is the good place