KR in Prolog

Prof. Mauro Gaspari
Dipartimento di Informatica Scienze e Ingegneria (DISI)
mauro.gaspari@unibo.it

Representing Knowledge with definite clauses

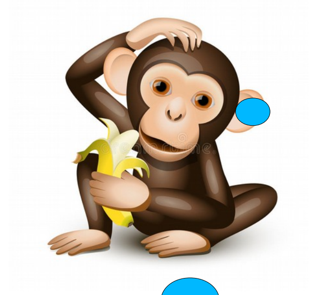
- The coronavirus special laws say that is a crime for Italian to leave their home city.
- John is an italian, he lives in Milan but now he moved to Puerto Escondido.

```
italian(john).
lives(john,milan).
moved(john,milan,puerto_escondido).
abroad(puerto_escondido).
abroad(london).
abroad(paris).
escaped(X):-lives(X,Y),moved(X,Y,Z),abroad(Z).
criminal(X):-italian(X),escaped(X).
```

Observations

- This is a toy example, but the representation can be made more adding more details.
- Adding more information:
    - Where people works.
    - Special needs
    - Medical needs.
- Representing time.

A new specification



- After the 11th of March 2020 an Italian coronavirus special law says that Italians can leave their home only if they are not quarantined and if one of the following motivations arises:
    - Work reasons.
    - Medical reasons.
    - Special needs.
    - Return to the place of residence.
- It is a crime not following these indications.

Monkey “gangs” have taken to the streets of Thailand

A notion of time

- The laws are valid after the 11th of March 2020, so we need to represent time. Time can be represented as a FOL term as follows:

  date(D,M,Y)

  where Day, Month, Year are integers.

- We need to define two predicates: after and before:

- before(DATE1,DATE2) holds i DATE1 is before DATE2.

```
before(date(D1,M1,Y1),date(D2,M2,Y2)):-Y1<Y2.
before(date(D1,M1,Y1),date(D2,M2,Y2)):-Y1==Y2,M1<M2.
before(date(D1,M1,Y1),date(D2,M2,Y2)):-Y1==Y2,M1==M2,D1<D2.
```

- after(T1,T2) holds if T1 is after T2.

```
after(date(D1,M1,Y1),date(D2,M2,Y2)):-Y1>Y2.
after(date(D1,M1,Y1),date(D2,M2,Y2)):-Y1==Y2,M1>M2.
after(date(D1,M1,Y1),date(D2,M2,Y2)):-Y1==Y2,M1==M2,D1>D2.
```

- Note that after is not the negation of before. We can use == to state that two dates are the same.

Predicates

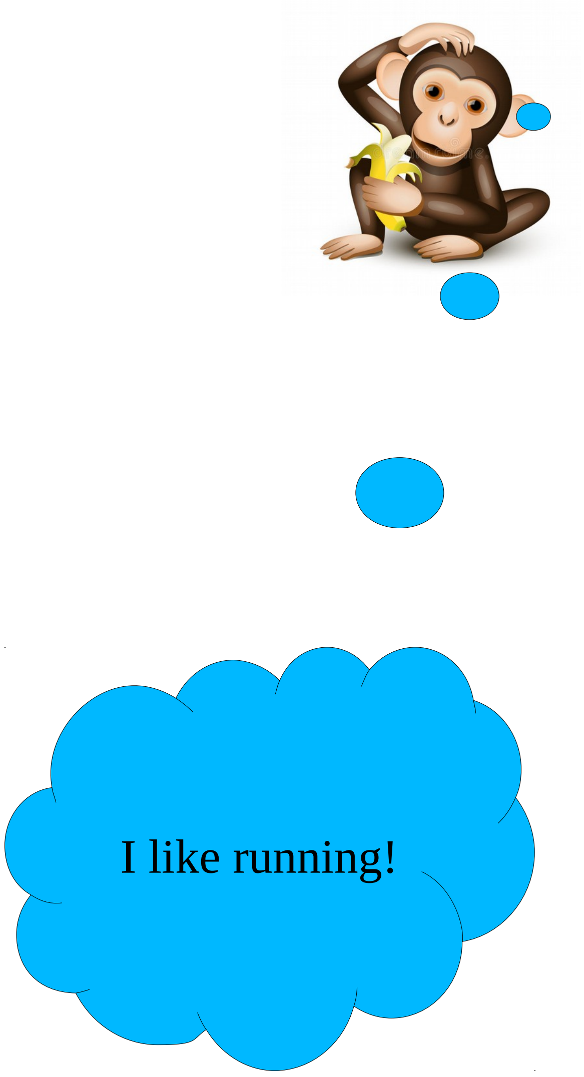
- works(person,place) ==> states that a person works in a given place.
- quarantined(person,date1,date2) ==> a person is under quarantine from date1 to date2.
- covid_positive(person,date1) ==> a person is positive to coronavirus from date1.
- lives(person,place) ==> it means residence.
- special_needs(person,date,place) ==> person has special needs at a given date in a place.
- healthcare(person,date,place) ==> a person has ealthcare needs at a given date.
- disease(person,dname,date) ==> a person has a diagnosis of a give disease from a given date.
- moved(person,from,to,date) ==> a person has moved from a place to another at a given date.
- dog(person,dname,date) ==> a person has a dog from a given date.
- today(date) ==> represents the date of today.

KB 1

```
illegal(X,D):-covid_positive(X,DC),after(D,DC).
illegal(X,D):-quarantined(X,DI,DF),after(D,DI),before(D,DF).


motivation(X,D,P):- special_needs(X,D,P).
motivation(X,D,P):- healthcare(X,D,P).
motivation(X,D,P):- works(X,P).


allowed(X,D,P):- motivation(X,D,P),lives(X,Q),moved(X,Q,P,D).
allowed(X,D,P):-  lives(X,P),disease(X,DS,DD),
                       after(D,DD),need_movement(DS).
allowed(X,D,P):- lives(X,P),dog(X,N,DD),after(D,DD).


criminal(X,D,P):- illegal(X,D).
criminal(X,D,P):-\+ allowed(X,D,P).
```

KB 2

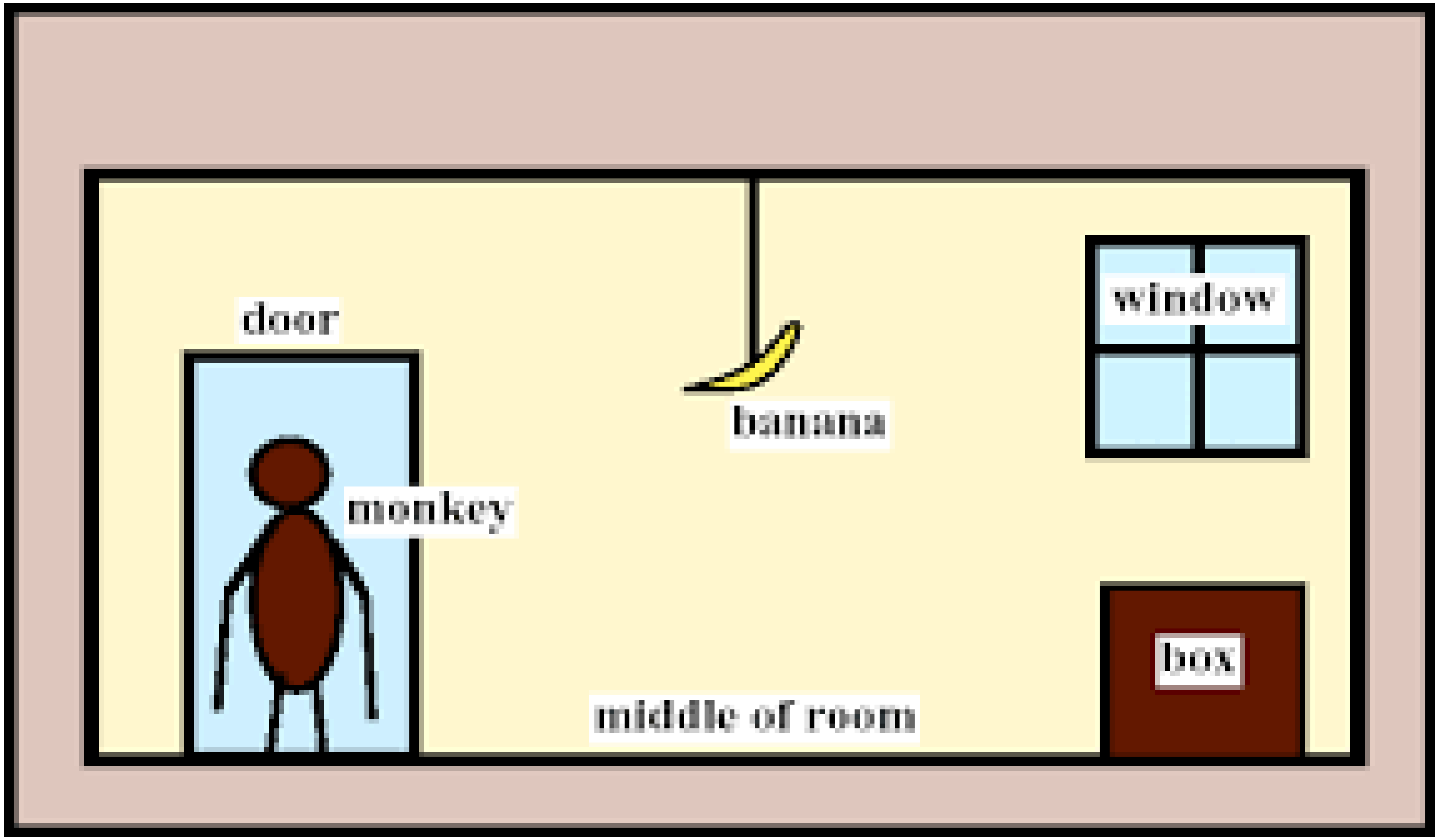
```
quarantined(paul,date(1,2,2020),date(1,3,2020)).
healthcare(peter, date(25, 2, 2020), venice).

special_needs(mary, date(25, 2, 2020), venice).

lives(john,milan).
lives(mary,bologna).
lives(diana,bologna).
lives(peter,bologna).
lives(paul,milan).

dog(mary,kurt,date(1,4,2017)).
works(john,venice).
moved(john,milan,venice,date(23,2,2020)).
moved(john,milan,venice,date(25,2,2020)).
moved(peter,bologna,venice,date(25,2,2020)).
moved(mary,bologna,venice,date(23,2,2020)).
disease(diana,diabets,date(21,3,2015)).
need_movement(diabets).
```



Observations

- Prolog is expressive enough for encoding realistic contents.
- The computation is efficient.
- However, the knowledge engineer should consider control issues in the designo of the KB.
- Only certain goals are allowed.

Monkey and Banana



Representing states

initial state: Monkey is at door,
Monkey is on floor,
Box is at window,
Monkey doesn't have banana.

state(Monkey location in the room,
Monkey onbox/onfloor,
box location,
has/hasnot banana)

I know how to do!

Legal Actions

```
do( state(middle, onbox, middle, hasnot),   % grab banana
    grab,
    state(middle, onbox, middle, has) ).

do( state(L, onfloor, L, Banana),           % climb box
    climb,
    state(L, onbox, L, Banana) ).

do( state(L1, onfloor, L1, Banana),         % push box from L1 to L2
    push(L1, L2),
    state(L2, onfloor, L2, Banana) ).

do( state(L1, onfloor, Box, Banana),        % walk from L1 to L2
    walk(L1, L2),
    state(L2, onfloor, Box, Banana) ).
```

Control

```
% canget(State): monkey can get banana in State
canget(state(_, _, _, has)).           % Monkey already has it, goal state

canget(State1) :-                      % not goal state, do some work to get it
    do(State1, Action, State2),        % do something (grab, climb, push, walk)
    canget(State2).                    % canget from State2

% get plan = list of actions
canget(state(_, _, _, has), []).       % Monkey already has it, goal state

canget(State1, Plan) :-                % not goal state, do some work to get it
    do(State1, Action, State2),        % do something (grab, climb, push, walk)
    canget(State2, PartialPlan),       % canget from State2
    add(Action, PartialPlan, Plan).    % add action to Plan

add(X,L,[X|L]).
```

Example

```
?- canget(state(atdoor, onfloor, atwindow, hasnot), Plan).
Plan = [walk(atdoor, atwindow), push(atwindow, middle), climb, grasp]
Yes

?- canget(state(atwindow, onbox, atwindow, hasnot), Plan ).
No

?- canget(state(Monkey, onfloor, atwindow, hasnot), Plan).
Monkey = atwindow
Plan = [push(atwindow, middle), climb, grasp]
Yes
```

I have got it!