PROJECT 2 REPORT

Sami Maizi < 77780 >

Supervised by Dr. Tajeddine Rachidi

Introduction

This Project aims at implementing Knowledge based logic using Swish Prolog by recreating the Wumpus game. The main objective is for the main player to kill the monster – Wumpus – while traveling a world made of square spaces. If the player falls in the Wumpus' space, he dies. He has to kill him while being in a neighboring space using indications from the environment.

Elaboration

The first step was to establish the necessary predicates:

- Room
- Breeze
- Pit
- Wumpus
- Stench
- Gold
- Adjacentto
- Safe
- MoveFromTo
- GrabGold
- ShootWumpus

These represent the essential building blocks of the world of Wumpus and information based on these are used to solve the game. The logical link between all this information is what makes our agent rational. As an example, the room predicate was implemented using a list that represents the set of rooms/spaces where we have been and where our agent can make assumptions about the surrounding rooms that are considered adjacent.

```
adjacent([X,Y],L) :- Xr is X+1, L=[Xr,Y].
adjacent([X,Y],L) :- Xl is X-1, L=[Xl,Y].
adjacent([X,Y],L) :- Yt is Y+1, L=[X,Yt].
adjacent([X,Y],L) :- Yb is Y-1, L=[X,Yb].
```

It is also important to make limits to the overall world by setting a border

```
border([X,Y]) :- X<1; X>4; Y<1; Y>4.
```

Here's the implementation of some predicates:

Pit:

```
pit([X,Y]) :- forall(adjacent([X,Y],L), (breeze(L);border(L))).
pit([X,Y]) :- adjacent([X,Y],L), visited(L), breeze(L), forall(adjacent
(L,L2),(L2 == [X,Y] ; psafe(L2) ; border(L2))).
```

And the implementation of the failure state (Wumpus or pit room):

```
fail_agent([X,Y]) :- pit([X,Y]); wumpus([X,Y]).
```

Make deduction/statement about environment

```
makestatement([X,Y]) :-
           forall((pit_location(PL),adjacent([X,Y],PL)), (format('there is a breeze in ~p~n',[[X,Y]]),assert(breeze([X,Y])))),
           forall((wumpus_location(L),adjacent([X,Y],L)), (format('there is a stench in ~p~n',[[X,Y]]),assert(stench([X,Y])))),
           forall((gold_location(G),([X,Y] == G)),(assert(glitter([X,Y])),score(S), N is S + 500 , format('I have found GOLD, Score is now ~p~n',[N]),
retractall(score(_)), assert(score(N)))).
```

Presence of the Wumpus

```
wumpus([X,Y]) :- forall(adjacent([X,Y],L), (stench(L);border(L))), retractall(wumpus_final_location(_)),assert(wumpus_final_location([X,Y])).
wumpus([X,Y]) :- adjacent([X,Y],L), visited(L), stench(L), forall(adjacent(L,L2),(L2 == [X,Y];wsafe(L2); border(L2))),
            retractall(wumpus_final_location(_)),assert(wumpus_final_location([X,Y])).
```

Presence of Gold

```
gold([X,Y]) :- glitter([X,Y]).
```

Other Variables/Predicates

```
failure(X):- wumpus_location(W), pit_location(P), (X=W;X=P), format('Eaten!') ,halt.
exist(X):- existgood(X);existmaybe(X).
start:-
    init,
    agent_location(AL),
    \+acte(AL),
    wumpus_final_location(Z),
    (Z=[-1,-1])->  format('The agent failed to find the Wumpus~nFAILED !~n'); (  wumpus_final_location(Z), format('The wumpus is located in ~p! I am
shooting my bullet!~n',[Z])),
    score(S),
    timer(T),
    format('Score: ~p~n',[S]),
    format('timer: ~p~n',[T]),
    format('WON!').

acte(X):-
    retractall(agent_location(_)),
    assert(agent_location(X)),
    update_score(-1),
    update_timer(1),
    format('I am in ~p~n',[X]),
    %failure(X),
    assert(visited(X)),
    makestatement(X),
    exist(L),
    get_next(N,L,X),
    wumpus_final_location(Z),
    Z = [-1,-1],
    acte(N).
```

```
get_next([X,Y],[X1,Y1],[X2,Y2]):-
    (adjacent([X1,Y1],[X2,Y2])) ->  ([X,Y] = [X1,Y1]);(adjacent([X1,Y1],L),visited(L)) ->([X,Y]=L).
update_score(X):-
    score(S),
    Z is S+X,
    retractall(score(_)),
    assert(score(Z)).
update_timer(X):-
    timer(S),
    Z is S+X,
    retractall(timer(_)),
    assert(timer(Z)).
```

Different Starting Configurations and experiments

init:-

retractall(timer(_)),

assert(timer(0)),

retractall(score(_)),

assert(score(30)),

retractall(gold_location(_)),

assert(gold_location([4,2])),

retractall(wumpus_location(_)),

assert(wumpus_location([3,5])),

retractall(pit_location(_)),

assert(pit_location([1,3])),

assert(pit_location([4,1])),

retractall(agent_location(_)),

assert(agent_location([1,1])),

retractall(wumpus_final_location(_)),

assert(wumpus_final_location([-1,-1])).

```
?- start.
I am in [1,1]
I am in [2,1]
there is a breeze in [2,1]
I am in [1,1]
I am in [1,2]
I am in [2,2]
I am in [1,2]
I am in [1,3]
there is a breeze in [1,3]
I am in [2,2]
I am in [3,2]
there is a breeze in [3,2]
I am in [1,3]
there is a breeze in [1,3]
I am in [2,3]
I am in [3,3]
I am in [2,3]
I am in [2,4]
there is a breeze in [2,4]
I am in [3,3]
I am in [4,3]
there is a stench in [4,3]
I am in [4,2]
I have found GOLD, Score is now 513
I am in [2,4]
there is a breeze in [2,4]
I am in [3,4]
there is a stench in [3,4]
I am in [4,2]
I have found GOLD, Score is now 1010
I am in [4,1]
there is a breeze in [4,1]
The wumpus is located in [3,5]! I am shooting my bullet!
Score: 1009
timer: 21
WON!
true.
```

init:-

```
retractall(timer(_)),
assert(timer(0)),
retractall(score(_)),
assert(score(30)),
retractall(gold_location(_)),
assert(gold_location([4,4])),
retractall(wumpus_location(_)),
assert(wumpus_location([0,2])),
retractall(pit_location(_)),
assert(pit_location([1,3])),
assert(pit_location([3,3])),
```

```
  assert(pit_location([4,1])),
  retractall(agent_location(_)),
  assert(agent_location([1,1])),
  retractall(wumpus_final_location(_)),
  assert(wumpus_final_location([-1,-1])).
```

```
?- start.
I am in [1,1]
I am in [2,1]
there is a breeze in [2,1]
there is a stench in [2,1]
I am in [1,1]
I am in [1,2]
there is a stench in [1,2]
I am in [2,2]
I am in [3,2]
there is a breeze in [3,2]
there is a breeze in [3,2]
there is a stench in [3,2]
I am in [2,2]
I am in [2,3]
there is a breeze in [2,3]
there is a stench in [2,3]
The wumpus is located in [0,2]! I am shooting my bullet!
Score: 22
timer: 8
WON!
true.
```

Limitations

There seems to be some cases where even if we find the Gold, the Wumpus is not found and killed, which is the main objective of the gain. This requires further investigation.

init:-

```
  retractall(timer(_)),
  assert(timer(0)),
  retractall(score(_)),
  assert(score(30)),
  retractall(gold_location(_)),
  assert(gold_location([4,2])),
  retractall(wumpus_location(_)),
  assert(wumpus_location([4,4])),
```

```
retractall(pit_location(_)),
assert(pit_location([1,3])),
assert(pit_location([3,3])),
assert(pit_location([1,4])),
retractall(agent_location(_)),
assert(agent_location([1,1])),
retractall(wumpus_final_location(_)),
assert(wumpus_final_location([-1,-1])).
```

```
?- start.
I am in [1,1]
I am in [2,1]
there is a breeze in [2,1]
I am in [1,1]
I am in [1,2]
I am in [2,2]
I am in [1,2]
I am in [1,3]
there is a breeze in [1,3]
I am in [2,2]
I am in [3,2]
there is a breeze in [3,2]
there is a breeze in [3,2]
I am in [1,3]
there is a breeze in [1,3]
I am in [2,3]
there is a breeze in [2,3]
I am in [3,2]
there is a breeze in [3,2]
there is a breeze in [3,2]
I am in [4,2]
I am in [4,3]
there is a breeze in [4,3]
there is a stench in [4,3]
I am in [4,2]
I am in [4,1]
there is a breeze in [4,1]
I am in [4,3]
there is a breeze in [4,3]
there is a stench in [4,3]
I am in [3,3]
I am in [3,4]
there is a breeze in [3,4]
there is a stench in [3,4]
I am in [2,4]
there is a breeze in [2,4]
I have found GOLD, Score is now 510
The agent failed to find the Wumpus
FAILED !
true.
```

Remediation: I could assume that a more evenly spaced world with less aggregated elements could solve this issue, the agent might fall in some sort of loop