Artificial Intelligence

Project 2

Save Westeros

22.11.2018

Alaa Kurdi	34-878	T-11
Ali Waleed	34-673	T-07
Ahmed Kamal	34-3110	T-11

Overview

A grid of size R × C that consists of R rows and C columns is given. The grid contains Jon Snow, White Walkers, Obstacles, a DragonStone and Free cells. Jon Snow can do the following actions:

- Move to a Free neighboring cell, a cell is considered Free if it does not contain a White Walker or an Obstacle.
- Collect DragonGlass if he is inside the DragonStone Cell.
- Kill all White Walkers in neighboring cells, but only if he has DragonGlass

Grid Generation

The problem grid is generated randomly using a java file GenerateGrind.java that takes the number of rows and columns of the grid and generates the grid in the form of facts in the knowledge base in an output file called "KnowledgeBase.pl".

The World is represented as a 2D array containing the following string representing each cell content

- Jon Snow Cell is Represented as "Jon
- DragonStone Cell is Represented as "DS"
- WhiteWalker Cells are Represented as "WW"
- Obstacle Cells are Represented as "O"
- Free Cells are Represented as "F"

The knowledge base file would contain the following:

The cell position would be represented as location(X, Y) where X represents the row number and Y represents the column number

1. rows(X): X is a valid row number; within height range.
2. columns(X): X is a valid column number; within width range.
3. jon(location(X,Y), 0, s0): represents the initial location of Jon Snow.
4. dS(location(X,Y)): represents the initial location of the DragonStone
5. whiteWalkers(location(X,Y) , s0): represents the location of each White Walker separately.
6. obst(location(X,Y)): represents the location for each obstacle separately.

Actions

Movement actions are represented as facts action(Name, DX, DY) where the Name would represent the direction of the motion, DX would be the change applied on the X to reach the new position with X being the original position, DY would be the change applied on the Y to reach the new position with Y being the original position.

NewX = OriginalX + DX

NewY = OriginalY + DY

The rest of the actions would be checked for inside the successor state axioms and checking for the possibility of the action inside the axiom.

Predicates Used

inBoundaries(location(X, Y)):-

Applys rows(X), columns(Y) which makes sure that location(X,Y) is inside the grid's boundaries.

freeCell(Cell, S):-

Checks if Cell does not contain an obstacle and a white walker; thus making sure that the cell is free and that Jon can move to this cell.

validMove(To, S):-

Checks if Jon can move to the cell labeled as "To" by checking if "To" is inside boundaries by calling the inBoundaries predicate and calling freeCell(To,S) to check if To is a free cell; if both conditions are true then Jon moving to this cell is a validMove.

goal(S):-

Checks if all White Walkers available in the initial situation are found dead in situation S.

Successor State Axioms

I. Jon Successor State Axiom

jon(location(X, Y), Dg, result(A, S)):-

- As a result of the action pickup the axiom would return true if Jon is in location(X,Y) in situation S and the DragonStone is in location(X,Y); thus Dg would be equal to the number of dragonglass collected from the DragonStone which means that Jon performed the pickup
- As a result of a moving action the axiom would return true if and only if Jon was at a previous location(XF,YF) in situation S and can make a valid move to location(X,Y).
- As a result of the killww action the axiom would return true if Jon in the previous situation S but if the DragonGlass in the previous situation was greater than 0 and there exists a White Walker or more in the neighboring cells.

II. White Walker Successor State Axiom

whiteWalkers(location(X,Y), result(A, S)):-

would return true if the White Walker in cell (X,Y) is alive in situation S as a result of actions north, south, east, west, pickup. However if the action done was killww, and Jon in state S was in a neighboring cell and has more than 0 Dragon Glass then the axiom would return false which represents that the White Walker dies.

Running Examples

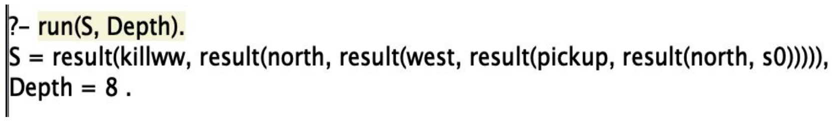
Put the GenerateGrid.java file in the same directory as finalMain.pl file,

In the GenerateGrid file change the variable fileName to the name of the file you want then update the grid file name in finalMain.pl line in the line ":- include('newFileName.pl').

Run the file finalMain.pl using prolog and call the predicate run(S, Depth). The output will be a sequence of moves.

I. Example 1

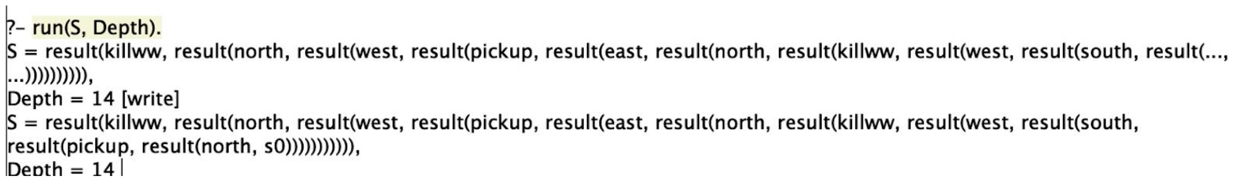
WW	F	WW
O	F	DS
F	F	Jon



```
?- run(S, Depth).
S = result(killww, result(north, result(west, result(pickup, result(north, s0))))),
Depth = 8 .
```

II. Example 2

WW	F	WW
O	F	DS
WW	F	Jon



```
?- run(S, Depth).
S = result(killww, result(north, result(west, result(pickup, result(east, result(north, result(killww, result(west, result(south, result(...,
...)))))))))),
Depth = 14 [write]
S = result(killww, result(north, result(west, result(pickup, result(east, result(north, result(killww, result(west, result(south,
result(pickup, result(north, s0))))))))))),
Depth = 14
```