Project 2 AI

13.01.2023

Amira Hossam El-Din Roshdy 46-1073

Shahd Medhat 46-1105

Mohanad Osama 46-2021

Overview

In this project we have implemented a simplified version of the project in milestone 1. We used notepad++ and SWI-prolog in order to implement our code. Project 2 is a simplified , logic based version in a world where we can have a maximum of 2 ships, each has only 1 passenger who doesn't expire. The grid is either 3x3 or 4x4.

All the specified data are in the knowledge base file “KB.pl” which is imported in our main file ‘ CG.pl ’ using “ :- consult('KB.pl'). ”.

Implementation

Main Functions

1. **goal(S).**

   This predicate returns a new state if the operator (right,left,up,down,pickup,drop) used on the input state is valid. It generates a plan, using the agent’s KB. The result of the query is a situation described as the result of doing some sequence of actions from the initial situation s0.

   This predicate is divided into two cases:

   - Case 1: If there are no more passengers to save, we send our coast guard agent to the station. We do this by first checking that the ships_loc predicate is empty then call the state predicate (explained later) to ensure that all passengers are dropped.

   - Case 2: There are still more passengers/ships. We initialize S as “result(drop,_)” because it's the last action that the coast guard agent will do and then we back-track. We then call the state predicate (explained later). The stopping condition is case 1.

2. state(M,N,X,Y,L,C,R).

This predicate creates a new state using successor state axioms on the current situation. It returns a new state with the corresponding correct operator according to the input state. This function keeps on concatenating the actions taken by the coast guard agent to a string result. The result is the backtracking of the actions that led to the goal state.

- The M ,N, X,Y ,L C,R represent the grid width, grid height, coast guard X-location, coast guard Y-location, list of ships locations, current picked up passengers and finally the result respectively.

- Move Up: X needs to be greater than or equal to 0, to move down; X needs to be less than M, to move left; Y needs to be greater than or equal to 0, and finally to move right; Y needs to be less than N.

- Pickup: we have to check that the agent is in the same cell as the ship remaining not saved. In addition to that, the coast guard agent needs to be carrying a number of passengers less than max capacity defined in the KB. Once a ship location has been visited and the agent has saved the passenger, we remove it from the list of ship_loc predicate. We also update the current picked up number of passengers for the new state and concatenate a "result(pickup,previousResult)" to the beginning of the set of output actions.

- Drop: the agent picked up passengers to be 0 and in the same cell as the station.

Helper Functions

1. canPickup(X,Y,RemainingPassengers,R):

This predicate checks if the agent can pick up a passenger at the current location. We first check if any remaining passengers exist in the current location by using the built in predicate member and if yes, we remove this ship from the ship list (using the removeShip predicate which is explained later).

2. removeShip(X,Y,[H|T],T):

This predicate removes a pair representing a location of X and Y from a list of ship locations.

- Base Case : if the position is the same as the first location in the list.

- Case 2 : recurse through the list to find the location by dividing the head into another head and tail then passing the new tail and end tail to the predicate again.

Test Cases (KB.pl)

I. Query 1 - goal(S).

```
9 ?- goal(S).
S = result(drop, result(left, result(pickup, result(right, result(drop, result(left, result(up, result(pickup, result(right, result(down, result(down, s0))
)))))))) .
```

II. Query 2 - goal(result(drop, result(up, result(left, result(pickup, result(down, result(right, result(drop, result(left, result(pickup, result(down, result(right, s0))))))))))).

```
30 ?- goal(result(drop, result(up, result(left, result(pickup, result(down, result(right,
|    result(drop, result(left, result(pickup, result(down, result(right, s0))))))))))).
true .
```

III. Query 3 - goal(result(drop, result(up, result(left, result(pickup, result(right, result(down, result(drop, result(left, result(pickup, result(right, result(down, s0))))))))))).

```
31 ?- goal(result(drop, result(up, result(left, result(pickup, result(right, result(down,
|    result(drop, result(left, result(pickup, result(right, result(down, s0))))))))))).
true .
```

IV. Query 4 - goal(result(up,result(down,s0))).

```
32 ?- goal(result(up,result(down,s0))).
false.
```