Introduction to Artificial Intelligence
Assignment 1: Compass and Pirates
Report

Vladislav Danshov, BS21-05

1. Assumption

1) Agent travels inside the field and cannot go out
2) Cost of moving in any direction by one cell is 1 point
3) When we generate map randomly, we set agent in (0,0) initial point

2. PEAS of Jack Sparrow

1) Performance measure – finding the shortest safe path to Dead Man's Chest
2) Environment – 9x9 cell map
3) Actuators – movement on map, visiting Tortuga, killing Kraken, taking Chest
4) Sensors – spyglass (2 types, depends on perception scenario), compass

3. Properties of the environment

1) Partially observable – Agent can observe only his perception zone using spyglass
2) Single agent – only one agent moves (Jack)
3) Sequential – decisions may has consequences
4) Discrete – Agent moves cell by cell
5) Dynamic – Agent can change the state of environment by killing Kraken
6) Partially known –Agent knows coordinates of Tortuga and Chest since he has compass, but he doesn't know coordinates of enemies and danger zones

4. Search Algorithms

4.1. Backtracking

In my backtracking implementation algorithm tries to go through all possible paths. My function calls itself for every neighbour cell. To take a neighbour cell and sore information about all cells I have my own Map class. Map class contains a 2d 9x9 array of Nodes. My class Node store coordinates and other information about cell (if it has any type of object or danger zone). What about algorithm itself, to improve it I decided to store the shortest path length to each cell in 2d array. From the beginning we fill this array by 81, since it is the maximum possible path length + 1. And if we visit cell where we have already been and current path grater then which written in array, so we exit from this branch of recursive algorithm. Else, we update the shortest path length value and continue algorithm by traversing all neighbors and taking the shortest path to destination. When we came to destination, algorithm returns full path from start to current node.

4.2 A* search

As we know from the lectures A* algorithm tries to reach the destination from the initial node as quickly as possible by checking the distance to final node. In comparison with Backtracking, A* algorithm may not to try all possible paths. To reach such efficiency I created a wrapper class AStarNode for Node that stores Node and additional information for A* algorithm such as parent Node and values f, g, h. Where g is path length from initial node to current one, h is the distance between current node and final node, and f = g + h. For A* implementation I used a PriorityQueue of AStarNode that contains in Java standard library. To make PriorityQueue works properly my AStarNode implements Comparable<AStarNode> and I overrode compareTo method, so that in head queue will be nodes with smaller f values, if f values are equal then we look at h values. So that we pick the element from the head of queue (element with the least cost). Then for current node we add all neighbors to queue. And then we add current to closed set. If we reach to destination, we stop the algorithm. Moreover, to improve A* algorithm I used the same technique as in Backtracking algorithm (store the shortest path length for each node).

5. Solution of the problem

For solving the problem (Finding shortest path from start to Chest) I try to compute the following for each algorithm:

1) Direct path to the Chest

2) Path to Tortuga (to have ability to kill the Kraken) plus path from Tortuga to Chest.

Moreover, If by 1st step we found path that length equals to distance between start and Chest, we can skip the 2nd step.

6. Maps

For displaying maps I use the following notation:

j – Jack Sparrow

d – Davy Jones

k – Kraken

r – Rock

c – Dead Man’s Chest

t – Tortuga

e – Enemy perception zone (danger zone)

0 – Empty cell

6.1 Maps that impossible to solve (Lose case)

When we generate maps there is a possibility to generate the map where Jack has no ability to reach the Dead Man’s Chest. Also, my Program has ability to generate such maps by corresponding query. Here you can see some of such maps:

0 1 2 3 4 5 6 7 8 0 j e e 0 r 0 0 0 0 1 e d e 0 0 0 0 0 0 2 e e e 0 0 0 0 e 0 3 0 0 0 0 0 0 e k e 4 0 0 0 0 0 0 0 e 0 1st map 5 0 0 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 0 0 7 0 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 t 0 c	0 1 2 3 4 5 6 7 8 0 j 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 t 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 2nd map 5 0 0 0 0 0 0 e e e 6 0 0 0 0 0 0 e d e 7 0 0 0 0 0 0 e e e 8 0 0 0 0 0 e r e c
0 1 2 3 4 5 6 7 8 0 j 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 t 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 e e e 3rd map 5 0 0 0 0 0 0 e d e 6 0 0 0 0 0 0 e e e 7 0 0 0 0 0 r e 0 0 8 0 0 0 0 0 e k e c	0 1 2 3 4 5 6 7 8 0 j 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 0 0 r 0 0 0 0 0 0 4th map 5 0 0 0 0 0 0 e e e 6 0 0 0 0 0 0 e d e 7 0 0 0 0 0 0 e e t 8 0 0 0 0 0 e r e c

1) In first case Davy kills Jack in the start cell
2) Here even if Jack kills Kraken he cannot reach Chest because of rock (Kraken is in rock)
3) Jack cannot kill Kraken since he can't be on diagonal cell with kraken because of rock
4) Jack cannot reach Tortuga and can't kill Kraken

7. Statistical analysis

Also my program can collect data to analyze and analyze collected data by corresponding queries. This analysis based on 1000 randomly generated maps. I test both algorithms on 2 perception scenarios.

Perception scenario	1		2	
Algorithm	Backtracking	A*	Backtracking	A*
Mean time (ms)	7.301.36	0.103695	18.011152	0.058578
Mode time (ms)	0.03	0.04	0.01	0.03
Median (ms)	7.189	0.06245	17.1996	0.0448

Standard deviation (ms)	9.892051	0.225478	20.435781	0.073216
Wins	918		934	
Loses	82		66	
Percentage of wins	91.8%		93.4%	
Percentage of loses	8.2%		6.6%	

By analyzing this table we can notice that A* spend much less time to solve problem then Backtracking in both perception scenarios. It is interesting that in second perception scenario time of backtracking algorithm is 2 times grater then in 1st one, but time of A* has been improved in 2nd scenario.

8. GitHub

All my source code for this assignment, its output files, statistics and this explanation are available on my GitHub repository by this link https://github.com/vladdan16/IntroToAI/tree/main/Assignment_1.