MULTIAGENT PATH FINDING

AIFA-ASSIGNMENT 1

Members

1. 16IM10012 – Lingala Girish
2. 17IM30001 – Abhinav Sonowal
3. 20RJ92R04 – Meheresh Masanpally

CENTRE OF EXCELLENCE IN ARTIFICIAL INTELLIGENCE

IIT Kharagpur,INDIA

PROBLEM STATEMENT:

Consider a robot-assisted warehouse management system. A group of robots must collect items from specified locations, deliver them to desired locations, and then return to their respective destinations. This can be viewed as a multiagent scheme with a common goal. Assume that the whole planning process will take place in a 2D grid of size n × m. Each location (Li) on the warehouse floor can be denoted by a pair of integers Li=(xi, yi). Each cell on the grid can be categorized in one of the following ways (see diagram below) - source location (P1, P2 ...), destination location (D1,D2,...), temporary storage location (TS1,TS2,....), obstacle (black square),normal (rest of the cells). Source & destination denotes pick-up and drop locations respectively.

The term "temporary storage location" refers to a location where robots can temporarily store objects. Obstacles are places where the robot is not able to travel. The rest of the cells are classified as regular.

Let there be k number of robots and r number of tasks. Given are the details of robot and tasks.
Let's take k=4 robots and r= 4 tasks for example:

ROBOTS	LOCATION	
	Initial	Final
ROBOT1	R1	E1
ROBOT2	R2	E2
ROBOT3	R3	E2
ROBOT4	R4	E3

TASKS	LOCATION	
	Pickup	Deliver
Task1	P1	D1
Task2	P2	D3
Task3	P3	D3
Task4	P4	D2

Assuming that a robot can only move one cell (vertically or horizontally) per time step, and that a normal cell can only hold one robot. Multiple robots can be accommodated at the same time at the source, destination, and temporary storage locations. Our goal is to build a work schedule that helps us to complete all tasks in the shortest period of time possible. We need to construct both optimal and heuristic algorithms.

We take a grid of size(m*n) and assign start locations and end locations of a robot and assign pickup and delivery locations of a task according to the problem, assign storage stations.
We can take any m and n values. Here, we've considered 6*10 grid.

R1			P2		TS2				
						E1			D3
		D3		P1				E3	
	R2		TS1						P4
	D2							D1	
R4		R3			E2	TS3		P3	

Here black colored boxes are blocked cells where no robot can go through that cell. TS texted are storage stations where robots can store. Blue colored are starts and end locations of robots and Red colored are pick-up and end locations of tasks.

The following assumptions are considered while solving the path finding problem.

- Each Agent can move a single step at a single unit of time.
- the static obstacles will not be moving.
- Dynamic obstacles need to be handled by the agents.
- The robots may use the temporary location not in all cases and will have priorknowledge about its usage.

Our Approach:

In our source code, we have used two input files: *input1.yaml* and *input2.yaml* and an output file named *output.yaml*.
Command to run the program: *python cbs.py input2.yaml output.yaml*

INPUT file:

It is the input file which we have to predefine the number of Robots which are available to complete the all tasks. We also have to predefine the Robot's start and end locations and Task's pickup and delivery locations.

For any agent, e.g. agent0, it follows the following format in our input file:

```
goal: [7, 2]
name: agent0
start: [2, 5]
```

It means agent0 starts from the start position and reaches to this goal position. Like this, we have the following four tasks as mentioned below.

Task1: robot start position to pick-up location.

Task2: pickup location to storage station.

Task3: storage station to delivery location

Task4: delivery location to final location of robot.

In this file, we have other inputs which are obstacle cells where robots cannot progress through that cell.

We have assumed two robots for simplicity in our input file but we can increase the number of robots. The grid size and the number of obstacles can be changed according to the user's needs. In assigning start location, end location, pickup location, end location, we have to be careful as these can't be obstacle cells.

OUTPUT FILE:

In the output file, every agent starts from their starting position at time t=0 secs to do the Task1 as we mentioned in the input file. Output is in the format such that at time t, xand y coordinates of a robot are printed as below. After completion of Task1 by all robots, Task2 will be initiated at time t=x where x is a time taken by the corresponding agent to complete its previous task i.e., Task1 here. Similarly, by completion of all tasks by all robots, we will get time taken by a robot to complete all of his tasks.

```
Agent0:          Agent1:
- t: 0           - t: 0
  x: 2             x: 3
  y: 5             y: 1
- t: 1           -t: 1
  x: 3            x: 3
  y: 5            y: 2
```

This means that at time t=0s,agent0 is at(2,5) and agent1 is at(3,1). At time t=1s,agent0 is at(3,5) and agent1 is at(3,2).
