Artifical Intelligence

Assignment 1 Report

1) PEAS Description of Agents
   (a) A drone that carries a packet from one location to another
   Type: Utility based reflex agent. To reach the goal the path is more important.
   Performance Measure:
   - It should obey some social and legal rules such as not flying at disturbing altitude, not flying over military areas and private property.
   - Not causing any accident by paying attention to other aircraft and high buildings.
   - Reaching the goal by following the rules
   - Calculating whether the fuel is enough or not for the distance to go

   Environment:
   - Buildings
   - Other air vehicles
   - Flying animals

   Actuators:
   - Propellers
   - Engine

   Sensors:
   - Camera
   - GPS
   - Altitude sensor
   - Lidar

   (b) A robot that sings songs until baby sleeps
   Type: goal based reflex agent, there is one goal: get baby to go to sleep
   Performance Measure:
   - Get baby to go to sleep for a certain period of time if there is not another problem with baby (on normal conditions)
   - The songs robot sings should be suitable for babies, not songs that will adversely affect the baby's subconscious.

   Environment:
   - baby
   - bed
   - toys

Actuators

- Speaker

Sensors:

- Camera
- Movement sensors
- Microphone

c) Activity recognition and anomaly detection software agent in an airport

Type: model based reflex agents. To recognize the normal or abnormal situations a trained model is required.

Performance measure:

- Recognition of normal and abnormal movements of aircrafts
- Detection of unsafe or dangerous movements of aircrafts
- Detection of whether an aircraft deviate from its landing and take-off routes

Environment:

- Pilots, Hostesses, passengers, seats, aircrafts

Actuators:

- Given signals across the abnormal situations

Sensors:

- Cameras, signals

d) An agent that classifies emails as spam or not

Type: goal based reflex agent. The goal is classifying emails spam or not.

Performance measure:

- Whether the emails identified as spam meet the spam criteria such as:
  - A large string of numbers in front of the @sign
  - Using free ISP
  - Emails with bad grammar, spelling or syntax might be spam
- Determine the official email as not spam

Environment:

- Servers, computers

Task env.	observable	deterministic	Episodic	Static	Discrete	Agents
a	Partially	Stochastic	Sequential	Dynamic	Continuous	Multi
b	Partially	Stochastic	Sequential	Dynamic	Discrete	Multi
c	Partially	Stochastic	Sequential	Dynamic	Continuous	Multi
d	Partially	Stochastic	Sequential	Semi	Discrete	Single

2) Admissible but Inconsistent Heuristics
   a) For admissible heuristic :
      h(n) <= h(n)*(true cost from n)
      h(n) >= 0 and h(G) = 0
      true cost of B to G is 9 so;
      0 <= h(n) <= 9

   b) f(C) = h(C) + g(C) = 5 + 2 = 7
      f(A) = h(S) + g(S) = 6 + 0 = 6
      f(B) = h(B) + g(B) = x + 1
      f(B) should not be greater than f(C) and should not be lower than f(A) so;
      5 and 6 values are valid for h(B)

   c) f(C) = h(C) + g(C) = 5 + 4 = 9
      f(D) = h(D) + g(D) = 7 + 2 = 9
      f(A) = h(A) + g(A) = 5 + 2 = 7
      f(B) = h(B) + g(B) = x + 1
      9 > x + 1 > 7 => x = 7 so h(B) should be 7

3) a) The problem is controlling the given agents whose aim is to reach their goal position in minimum amount of time steps in a maze.
   - Initial state: the agents are placed at a maze, the goal locations are given.
   - Actions: Move or pass. All agents can move up/left/down/right or can pass the move. There are two type of movement: valid or invalid. Invalid moves are moving onto walls, moving to the same square or waiting at the same square at

the same time for multiple agents, moving toward each other from between squares for multiple agents. The other defined movements are valid.

- Transition model: initial state, transition state: combination of possible moves for all agents, end state: all agents reach at goal position.
- State space: locations of agents result of combination of possible moves.
- Goal test: Checking whether the agents at their goals or not.
- Path cost: one less of number of nodes from start to end.
- Optimal solution is reaching goal positions in minimum amount of time steps

b) For maze 1 (input_1.txt) :

Results of A :*

number of generated nodes: 46

number of expanded nodes: 31

max. number of nodes kept in the memory (equal to generated in my implementation) : 46

runtime: 0.11 sec

Results of BFS:

number of generated nodes: 50

number of expanded nodes: 44

max. number of nodes kept in the memory (equal to generated in my implementation) : 50

runtime: 0.12 sec

Results of DFS:

number of generated nodes: 34

number of expanded nodes: 14

max. number of nodes kept in the memory (equal to generated in my implementation) : 34

runtime: 0.10 sec

In this problem A* and BFS algorithm can find the shortest path. DFS algorithm can find a solution but does not guaranteed the shortest path. For this maze runtime of DFS is shortest but it is not a measure for all mazes.

DFS algorithm starts from root and goes as far as it can down in this path. When there is no child node (no different movement for current path in this problem) it go back the last visited parent and go on until there is no visited node. In this problem we starts from given position and goes on a path until path ends. If the path is so long, nodes kept on the memory is increased a lot and the running time for find the solution can took so long. This algorithm is not a good way for this problem.

BFS algorithm starts from root and expand all the nodes depth by depth. For example after the root expanded, all children of the root are expanded in order. So this algorithm can find shortest path. However, if the path is so long and the possible paths are so much, the run time of the solution can took so long. Also this algorithm is not a good way for this problem.

d) For maze 3:

Results of A :*

number of generated nodes: 2933

number of expanded nodes: 20

max. number of nodes kept in the memory (equal to generated in my implementation) : 2933

runtime: 10 sec

A* algorithm tries to find the path that have lowest-cost from start to goal. After start state (root) expands the node have lowest f value is expanded. In my implementation, if the f values are same, first tie break is looking at heuristic values. The second is expand the latest generated node. F is equal to sum of h and g when g is the step cost (length of the path from start to current node) and h is equal to max h value of the agents. h value of an agent is calculated by Manhattan distance from current square to goal square.

Talha Çomak

150160726