Homework 1

Aseem Raina — araina
CSC 520 - Artificial Intelligence 1

February 8, 2018

Question 1.

Answer. Agent for the Hunt Library

(a) PEAS Specification

Performance Measure: Reach destination B, don't bump into obstacles, box safety, distance to reach destination B, surrounding safety

Environment: stationary obstacles, rooms, doors, possible people in library, stairs, elevator doors

Actuators: Pistons, Arm for lifting objects, electric motors (for wheels), servos, piezoelectric actuator for sensing

Sensors: LIDAR or Proximity sensor, Piezoelectric sensor, Infrared sensor for possible line tracing, compass for orientation, GPS, camera

(b) For efficient function, it is not sufficient for the agent to be simple reflex. This is because the problem statement revolves around getting from a position A to be B in a dynamic environment, where obstacles are unknown. Suppose the agent encounters obstacle 'o' more than once, it should realize that it is not on the correct path in all probability, possibly moving in a loop. Utility level is preferred as it considers the happiness factor also, such that even if the box reaches the destination, the boxes should not break.

(c) In this case, the ability to move randomly will not help the agent performance. The Hunt Library environment is fairly complex, and the obstacles are unknown to the agent. The best will be to rely on sensor feedback to navigate. If the agent has the ability to move randomly, there is a high probability that will move around in loops, or waste valuable time in moving around the same place or a wrong path. A random move will help in the condition where multiple equally productive paths are possible, but that will be in a minority of cases. Also, random movement will help in mapping the environment initially so that the locations of the obstacles can be known.

(d) More sensors can be added to improve the feedback at every step. Eg. A piezoelectric sensor can be used to sense changes in pressure, so as to prevent the box from falling. A disadvantage is increased overhead, and more processing power needed to handle the sensor feedback. Also, sensor can provide noisy and incorrect data.

□

Question 2.

Answer. For each of the following agents, state the characteristics of its environment in the given table:

Agent	Fully/Partially Observable	Single/ Multi-Agent	Deterministic / Stochastic	Episodic/ Sequential	Static/ Dynamic	Discrete/ Continuous	Known/ Unknown
Tax Planner	Fully	Single	Deterministic	Episodic	Static	Discrete	Known
Search and rescue robots	Partially	Multi-Agent	Stochastic	Sequential	Dynamic	Continuous	Known
Document classifier	Fully	Single	Deterministic	Sequential	Static	Discrete	Known
fitness coach	Partially	Single	Stochastic	Sequential	Dynamic	Discrete	Unknown
Packet router	Partially	Multi-Agent	Stochastic	Episodic	Dynamic	Discrete	Known

Tax Planner
Fully observable as agent has access to all information in the environment (income history) required for planning.
Single as there is only one agent involved.
Deterministic as we can perfectly calculate the tax amount given the action and previous state.
Episodic if we consider tax planner as a means to calculate tax for a particular period. The income can be assumed to be constant, and the tax slabs and rate are fixed.
Static, as there is no change in the environment.
Discrete, as time intervals are defined. The environment is Known as the agent knows the rules that govern tax. For eg 8% state tax, 10% income tax on income above $10000 per month etc.

Search and Rescue robots
Partially observable as the environment in a search and rescue operation is not known, we don't know the location of causalities and people waiting to be rescued.
Multi-agent as in a search and rescue operation, multiple robots generally interact with each other, and their actions are dependent on each other. One robot should not dump rubble where another robot is cleaning that area.
Stochastic, since the environment is partially observable, and the next stage cannot be predicted perfectly.
Sequential, since the robot will require memory of past action to better tackle the next action.
Dynamic, since the environment can change at any time in case of a sudden fire or blast, or some structure falling.
Continuous, as time intervals are not fixed.
Known, as in a properly programmed robot (agent), it knows what rules govern the environment, and what action to take if a alive/dead person is found.

Document Classifier
Fully observable as each and every content of the environment (document/s) is known to the classifier (agent).
Single, as only one agent (classifier) is involved.
Deterministic, if we consider a classifier with 100% accuracy. If we consider some error, it can be stochastic.
Sequential, since a classifier is based on Machine Learning, and will require knowledge of previous state or a training data set to predict the next.
Static as the document or environment is not changing.
Discrete, as time interval is defined and fixed.
Known, as the classifier works on certain rules, line Naive Bayes classifier works on Bayesian probability.

Fitness Classifier
Partially observable as the internal state and fitness level of the body is not known to the agent.
Single, as only one agent is involved.
Stochastic, as the fitness agent can never accurately predict the probability of outcomes. The person using it can tire early, or there might be some accident.
Sequential, since the past action (exercise) can affect future best action (stop or continue).
Dynamic, since the environment (state of body), tiredness level can change.
Discrete, as normally a certain action or exercise routine is normally done for a specific interval of time.
Unknown, since the fitness agent is not advanced enough to deal with abnormal situations like person not completing exercise, or an accident.

Packet Router
Partially observable as the network setup and configuration is not known initially while routing packets.
Multi-Agent, as multiple such routers are present in the environment.
Stochastic as routing is not perfect, and rather than using the previous values of the agents (routers) in the environment, it uses the routing values from other routers in the current state. The router can never perfectly predict the next state or perfect path. There can be some sudden congestion. If we consider that the router updates the routing table using previous information or values, then it can be called as deterministic.
Episodic, since the router makes a decision based on the current state and not past configuration of network.
Dynamic, since the network environment is constantly changing based on routing decisions.
Discrete, as the location and time interval is fixed.
Known, as the router knows what decision to take if an event occurs. eg. stop routing from a certain path if over-congestion occurs.

□

Question 3.

Answer. Code submitted

(a) DFS

Expanded nodes: ['albanyNY', 'buffalo', 'calgary', 'chicago', 'dallas', 'denver', 'desMoines', 'indianapolis', 'kansasCity', 'minneapolis', 'montreal', 'omaha', 'portland', 'provo', 'santaFe', 'saultSteMarie', 'seattle', 'toledo', 'toronto', 'tulsa', 'vancouver', 'wichita', 'winnipeg']

Path: ['minneapolis', 'winnipeg', 'saultSteMarie', 'toronto', 'buffalo', 'cleveland']

BFS

Expanded nodes: ['buffalo', 'calgary', 'chicago', 'denver', 'desMoines', 'kansasCity', 'minneapolis', 'omaha', 'saultSteMarie', 'seattle', 'toledo', 'toronto', 'vancouver', 'wichita', 'winnipeg']

Path: ['minneapolis', 'winnipeg', 'saultSteMarie', 'toronto', 'buffalo', 'cleveland']

(b) DFS outperforms BFS in terms of number of nodes expanded if we consider the path from Provo to Calgary.

python2 SearchUSA.py bfs provo calgary Expanded nodes: ['atlanta', 'austin', 'chicago', 'cleveland', 'dallas', 'dayton', 'denver', 'desMoines', 'elPaso', 'ftWorth', 'houston', 'indianapolis', 'kansasCity', 'lakeCity', 'memphis', 'mexico', 'minneapolis', 'newOrleans', 'omaha', 'phoenix', 'provo', 'sanAntonio', 'sanDiego', 'santaFe', 'tallahassee', 'tucson', 'tulsa', 'wichita', 'winnipeg']

Number of Expanded nodes: 29
Path: ['provo', 'denver', 'wichita', 'omaha', 'desMoines', 'minneapolis', 'winnipeg', 'calgary']
Length of solution path: 7

python2 SearchUSA.py dfs provo calgary
Expanded nodes: ['austin', 'dallas', 'denver', 'elPaso', 'houston', 'japan', 'lasVegas', 'losAngeles', 'medford', 'modesto', 'phoenix', 'pointReyes', 'portland', 'provo', 'reno', 'sacramento', 'saltLakeCity', 'sanAntonio', 'sanDiego', 'sanFrancisco', 'sanJose', 'sanLuisObispo', 'santaFe', 'seattle', 'tucson', 'vancouver']
Number of Expanded nodes: 26
Path: ['provo', 'denver', 'dallas', 'houston', 'austin', 'sanAntonio', 'elPaso', 'tucson', 'phoenix', 'sanDiego', 'losAngeles', 'lasVegas', 'saltLakeCity', 'portland', 'seattle', 'vancouver', 'calgary']
Length of solution path: 16

(c) DFS will never outperform BFS in terms of length of solution path as BFS always returns the shortest path. DFS and BFS will return the same path if the vertices are connected in a single chain alphabetically. eg path between Memphis and Tallahassee.

(d) A* will never find a different path from A to B and B to A as the graph is undirected, so the best first path is the most optimal one in both directions as the weight is the same.

(e) DFS will never outperform A* in terms of nodes expanded as A* always choses the best path using a priority queue and DFS follows alphabetically.

□