L1F21BSCS0783 AI — Diyan Saeed

ASSIGNMENT 1

Task 1

(a)

<R, M, C>

R represents river side — starting (1), ending (2)
M represents missionaries
C represents cannibals

Two sides of river

<1, 3, 3> <2, 0, 0>] Initial state

↓

<1, 3, 1> <2, 0, 2>

↓

<1, 3, 2> <2, 0, 1>

↓

<1, 3, 0> <2, 0, 3>

↓

<1, 3, 1> <2, 0, 2>

↓

<1, 1, 1> <2, 2, 2>

↓

<1, 2, 2> <2, 1, 1>

↓

<1, 0, 2> <2, 3, 1>

↓

<1, 0, 3> <2, 3, 0>

↓

<1, 0, 2> <2, 3, 1>

<1, 0, 1><2, 3, 2>

<1, 0, 0><2, 3, 3>

Goal state

(b) we can use
Search Algorithm ↓ for this problem is Breadth First Search (BFS) because it guarantees the shortest path. Checking for repeated states is necessary to avoid infinite loop in certain search algo. BFS avoid revisiting state and exploring each state once and offers shortest path.

(C) The difficulty lies in the constraints and to think ahead to avoid leaving missionaries outnumbered by cannibals on either side. It requires logical thinking, planning and attention to detail to ensure that solution is valid

on each step.

TASK 2

Playing Soccer :-

Performance :
Score more goals than opponent in time
(Adversarial, Partially observable)

Environment :-
Field, opponents, teamates, ball, weather
(Dynamic, Continuous)

Actuators :-
Kicking, running, jumping
(continuous)

Sensors :-
Vision, hearing, ball location, body positioning and movement.

Exploring the Subsurface oceans of Titan:

Performance :-
Scientific data about the composition and potential life forms.
(Non-Adversarial, Partially observable)

Environment :-
High pressure, low temp. unknown chemicals and life forms
(Extreme, Unknown)

Actuators :-
submersible vehicle, instruments for collecting data.
(Discrete)

Sensors :-
cameras, Sonar chemical, pressure sensors.
(Limited)

Shopping for used AI Books :

Performance :-
Find and purchase
(Non-Adversarial, Partially observable)

Environment :-
Online marketplace, price and description
(Dynamic, Discrete)

Actuators :-
Typing search queries, clicking buttons,
(Discrete)

Sensors :-
Textual info.
(Limited)

Playing a Tennis Match :-

Performance :-
Score ~~Point~~ & more points to win
(Adversarial, ~~Pari~~ observable)
Partially

Environment :-
Tennis court, opponent, balls
(Dynamic, continuous)

Actuators :-
Swinging racket
(continuous)

Sensors :-
Vision
(Proprioception)

Practicing Tennis ball against a wall :

Performance :-
Improved skills
(Non-Adversarial, Fully observable)

Environment :-
Wall, balls, ~~racket~~
(static, Discrete)

Actuators :-
racket, moving around
(continuous)

Sensors :- Vision
(Proprioception)

Performing a High Jump :-

Performance :-
Jump over a bar
(Non Adversarial, Fully observable)

Environment :-
Open Space, runway, bar
(Static, Discrete)

Actuators
Running, Jumping
(Discrete)

Sensors Vision
(proprioception)

Knitting a Sweater :-

Performance :-
Design, Size
(Non-Adversarial, Fully observable)

Environment :-
Knitting needles, yarn, pattern
(static, Discrete)

Actuators
Hands using needles and yarn
~~(Non-adversarial, Fully observable)~~
(Discrete)

Sensors :- Vision
(Touch)

Bidding an Item :-

Performance :-
Win auction for desired item in lowest price
(Adversarial, Partially observable)

Environment :-
Action platform, auctioneer, bidder
(Dynamic, Discrete)

Actuators :- Entering bidding Amounts
(Discrete)

Sensors:

Textual information (Limited)

Task 3

1) Factors

Heuristic Design :- The chosen heuristic function might prioritize specific tile movement, leading to a path that reaches a seemingly good configuration but not optimal.

Example :- A heuristic focusing on minimizing might lead to state where one tile need swapping but it might block further progress.

Initial State :- If the initial ~~state~~ random starting configuration is challenging ~~the~~ it can get stuck regardless of heuristic

Deterministic moves :-

If the robot always prioritizes a specific "best" neighboring state based on heuristic, it might miss alternative paths that leads to better solution.

2) Modification for Improvments :-

Introduce Randomness :-

Implement a small chance of the robot exploring a non-optimal neighboring state even if it has a lower heuristic score.

Restart Strategy :-

Implement a mechanism to restart the search process from a new random configuration if the robot gets stuck for certain number of moves.

3) Methods

- **Number of misplaced tiles :-** This is a basic measure of how far the current state is from goal.

- **Distances from Goal Positions :-** This penalizes states where tiles are not only misplaced but also far from correct position.

- **Empty tile Position :-** Considering the position of the empty tile relative to misplaced tiles can help prioritize moves that open up pathways for further movement.