Intro to AI, Assignment 2

Jaskaran Singh Luthra (110090236)

October 2022

1   Question 1.

Three missionaries and three cannibals are on one side of a river, along with a boat that can hold one or two people. Find a way to get everyone to the other side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place. This problem is famous in AI because it was the subject of the first paper that approached problem formulation from an analytical viewpoint

a) Formulate the problem precisely according to Ch.3.1.11. Draw a diagram of the complete state space.

b) Implement and solve the problem using an BFS, DFS. Coding required. After you are done this question, please record your computer screen with audio to show the code, run your code, and explain your solution.

c) Design a heuristic function for this question and explain your design.2 Then implement and solve the problem using the greedy best-first search and the A* search algorithm. Coding required. After you are done this question, please record your computer screen with audio to show the code, run your code, and explain your solution.

d) Is it a good idea to check for repeated states based on your coding experience in this question?

a) Formulate the problem precisely according to Ch.3.1.11. Draw a diagram of the complete state space.

Answer

The goal of the puzzle is to move all three missionaries and three cannibals to the right side of the boat, which can only hold a maximum of two people at once.

A tuple **(m, c, s)** is used to represent each state.
Where,
• m for the number of missionaries,
• c for the number of cannibals, and
• s for the side of the boat

Start State $= (3, 3, 1) = 3m$, 3c and boat on left side
Goal State $= (0, 0, 0) = 0c$, 0c and boat on the right side

Action states:
(M,C) = (0,1), (1,1), (1,0), (0,2), (2,0)

Constraints:
• To get everyone to the other side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place.
• A boat can carry 1 or 2 people only at a time.
• At least one person should be on the boat to cross the river.

State Space Diagram for Missionaries and Cannibals problem is given below

ONE SIDE	RIVER	OTHER SIDE
mmm ccc B (3,3,1)		000 000 (0,0,0
mm cc (2,2,0)	1m 1c	m c B (1,1,1)
mmm cc B (3,2,1)	1m	c (0,1,0)
mmm (3,0,0)	2c	ccc B (0,3,1)
mmm c B (3,1,1)	1c	cc B (0,2,0)
m c (1,1,0)	2m	mm cc B (2,2,1)
mm cc B (2,2,1)	1m1c	m c (1,1,0)
cc (0,2,0)	2m	mmm c B (3,1,1)
ccc B (0,3,1)	1c	mmm (3,0,0)
c (0,1,0)	2c	mmm cc B (3,2,1)
m c B (1,1,1)	1m	mm cc (2,2,0)
0,0,0	1m 1c	mmm ccc B (3, 3, 1)

Figure 1: State Space Dig for Missionaries and Cannibals problem

b) and c)

Video Recording Link: https://uwin365-my.sharepoint.com/:v:
/g/personal/luthraj_uwindsor_ca/EfTjBxB1IxRIlHA4mfSaG3IBx6lzlUsebifIEDNts9iP5Q?
email=danwu%40uwindsor.ca&e=ffOOM1

Link for the Code: https://github.com/jksingh07/AI-Assignment

d)

Checking for repeated states while putting an algorithm into practise is unquestionably a smart idea. If states are not taken into account, the algorithm will repeat those states, wasting time and resources. The amount of memory needed is a crucial factor to take into account when keeping track of states. Any implementation will use all the RAM if there are an endless number of states that can be reacted to. For such implementations, it is necessary to erase the inaccessible states from the current state using a practical method.

2 Question 2.

Which of the following are true and which are false? Explain your answers briefly. For the qustions that answer false, you could consider providing a counterexample.

a) Depth-first search always expands at least as many nodes as A* search with an admissible heuristic

Answer a)

False

A fortunate DFS may expand exactly d nodes to accomplish the task to reach the goal state. According to the issue and the random possibility. In some situations, usually in straightforward ones where the goal state is near to the initial state, DFS can reach the goal state without extending more nodes than A*. DFS grows more nodes than A* in the majority of typical use cases where this is not the case.

Any graph-search method that is guaranteed to find the best answers will be dominated by A.

b) h(n)=0 is an admissible heuristic for the 8-puzzle.

Answer b)

True

h(n) must be less than the actual cost from vertex n to the target in order to satisfy this condition. That criterion will never be missed by h(n) = 0.

h(n)=0 A goal node's remaining optimum distance can never be overestimated. An admissible heuristic will never overstate the cost of getting to the desired state. The current state is the goal state if getting there costs nothing (0).

c) A* is of no use in robotics because percepts, states, and actions are continuous.

Answer c)

False

Distinct states can be created by abstracting the adjacent states. The system's computing power determines how far apart the states are from one another.

d) Breadth-first search is complete even if zero-step costs are allowed.

Answer d

True

The goal state will be attained and the state space will be considered complete if it is finite and has a defined depth (d). because if a goal exists, it will be found in O(bd) steps and occur at a finite depth d.

3 Question 3.

Trace the operation of A* search applied to the problem of getting to Bucharest from Lugoj using the straight-line distance heuristic. That is, show the sequence of nodes that the algorithm will consider and the f, g, and h score for each node. It is best to draw diagram similar to Figure 3.18 for tracing.
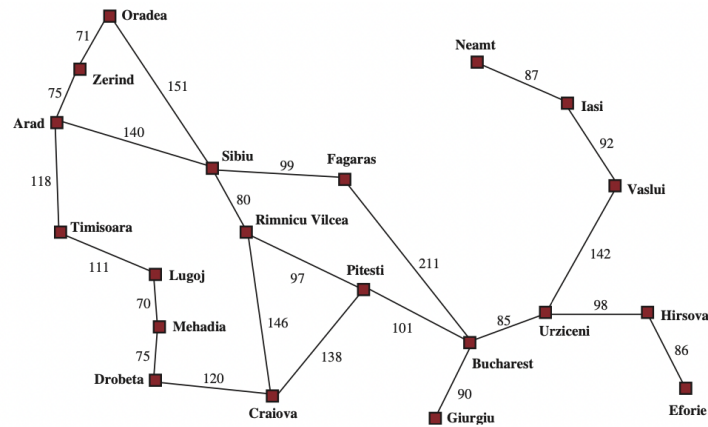
Given below is the roadmap of Romania

Figure 2: Roadmap of part of Romania.

Answer

A* Search

The most common informed search algorithm is A* search (pronounced "A-star search"), a best-first search that uses the evaluation function

f(n)=g(n)+h(n)

where g(n) is the path cost from the initial state to node n, and
h(n) is the estimated cost of the shortest path from n to a goal state, so we have
f (n) = estimated cost of the best path that continues from n to a goal.

Intial State: Lugoj
Goal State: Bucharest
Given below is the stages of A* search for Bucharest.

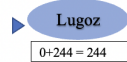|			
Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

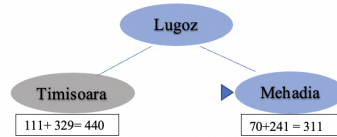Figure 3: Values of heuristic function h(n) - straight line distance to Bucharest.

Nodes are labeled with $f = g + h$
The h values are the straight-line distances to Bucharest taken from Figure 1.
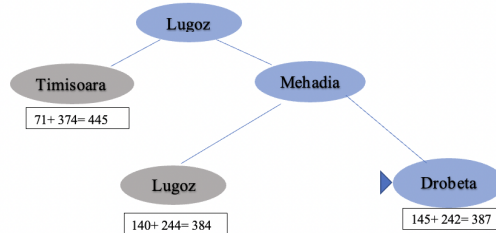
(a) The initial state

Lugoz
0+244 = 244

(b) After expanding Lugoz

Lugoz

Timisoara
111+ 329= 440

Mehadia
70+241 = 311

(c) After expanding Mehadia

Lugoz

Timisoara
71+ 374= 445

Mehadia

Lugoz
140+ 244= 384

Drobeta
145+ 242= 387

d) After expanding Drobeta

Mehadia		Craiova
290+ 241= 531		265+ 160= 425
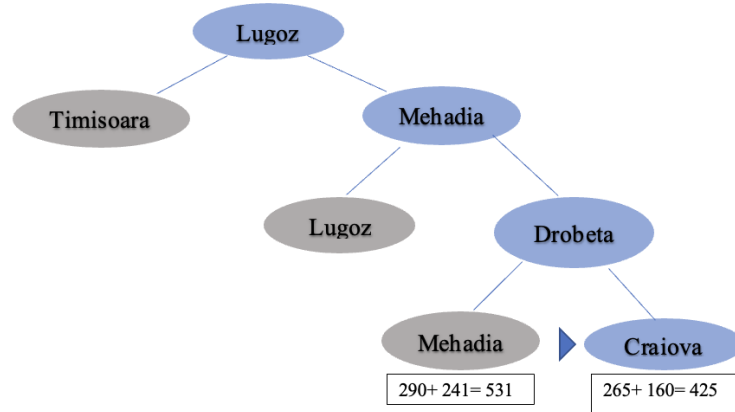
e) After expanding Craiova

Rimnicu		Pitesti
411+ 193= 604		403+ 100= 503
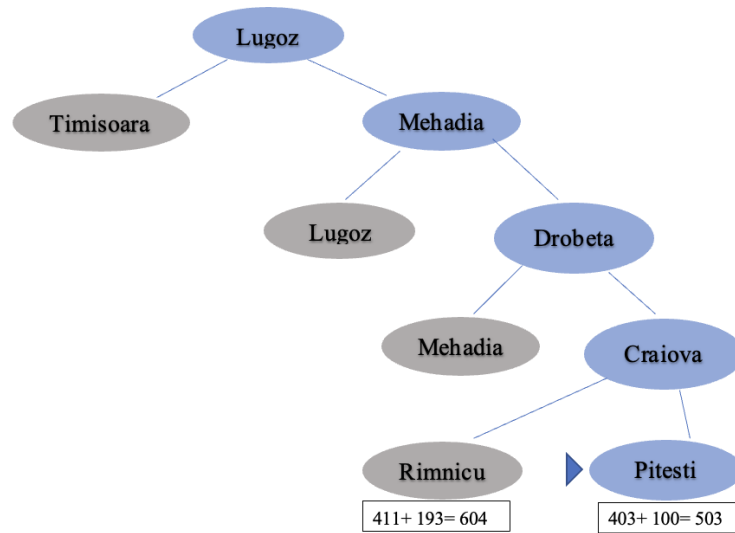
In the fig below, we reached the goal state Bucharest using A* search.

f) After expanding Pitesti

4 Question 4

For the A* search algorithm, discuss its completeness and cost optimality with respect to the heuristic function it uses under the followi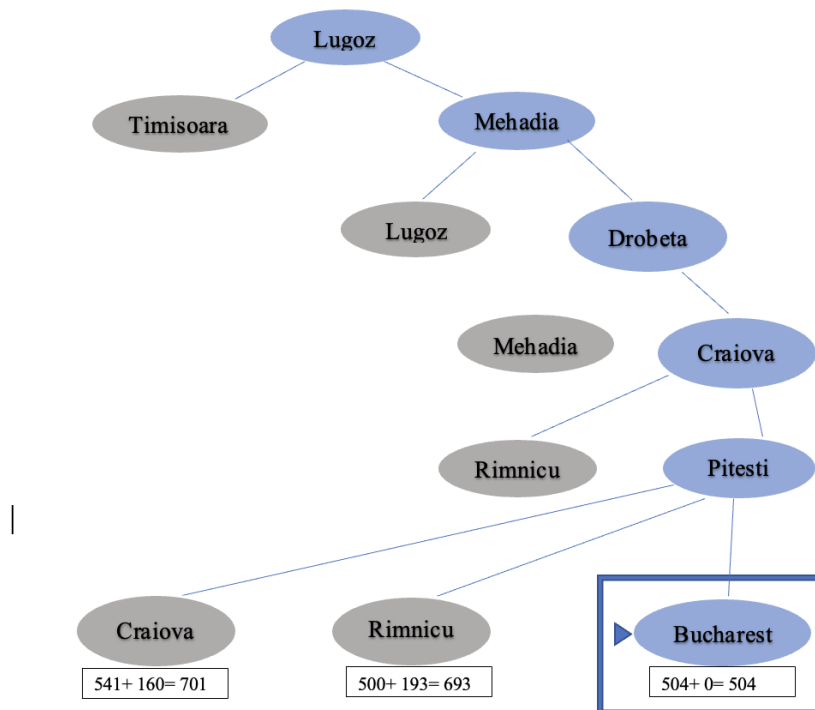ng conditions. You should use examples to support your claim. You could either design your own example or use the state space in Fig.3.1 or portion of it.

    a) The heuristic function used is admissible.
    b) The heuristic function used is always overestimate for each state.
    c) The heuristic function used sometimes overestimates for some states and sometimes underestimates for some states.

Answer
A* Search Algorithm evaluation function,

f(n)=g(n)+h(n)
g(n) is the path cost from the initial state to node n, and
h(n) is the estimated cost of the shortest path from n to a goal state, so we have
f (n) = estimated cost of the best path that continues from n to a goal.

a)The heuristic function used is admissible.
Every time a valid heuristic yields a cost, it is lower or equal to the real cost. Considered in the context of the example in 3.1 from figure 3.16 for the heuristic costs, it is evident that A* provides the best path while using the fewest nodes possible. Therefore, if a valid heuristic is used, the A* will be complete.

b)The heuristic function used is always overestimate for each state.
A heuristic for overestimating costs will always overestimate the cost for each node. Due to the uniform overestimation of costs across all nodes, A* will again be complete and optimal in this scenario. If we take the illustration at 3.1. Every node's cost values in figure 3.16 will increase consistently. This is true regardless of cost overestimation. The ideal path will also be provided because the relative path costs are unchanged.

c)The heuristic function used sometimes overestimates for some states and sometimes underestimates for some states
An ideal answer will not be provided by a heuristic that overestimates and underestimates for various states. The entries in figure 3.16 will be changed without any consistency if we take case 3.1 into consideration. The result will be a complete but not ideal solution.

5 Questiion 5

In the textbook, Ch.4.1.1 (Hill-climbing search), when describing hill-climbing search algorithm, it says
Starting from a randomly generated 8-queens state, steepest- ascent hill climbing gets stuck 86percent of the time, solving only 14percent of problem instances. On the other hand, it works quickly, taking just 4 steps on average when it succeeds and 3 when it gets stuck—not bad for a state space with 88 approx 17 million states.

a) Implement steepest-ascent hill climbing for the 8-queens problem, run it 1000 times, record the steps taken in each run, then calculate on av- erage your implement takes how many steps when it succeeds, and how many steps when it fails. Also calculate among the 1000 runs, how many times it succeeds to find a solution and how many times if fails to find a solution.

b) Implement steepest-ascent hill climbing with sideways move, for the 8-queens problem, run it 1000 times, record the failure/success rate; record the steps taken in each run, then calculate on average your implement takes how many steps when it succeeds, and how many steps when it fails.

Answer a) and b)

Putting eight queens on an 8 × 8 chessboard without any of them attacking one another is known as the "eight queens problem" (no two are in the same row, column, or diagonal). In a broader sense, the n queens problem involves putting n queens on an nxn chessboard.

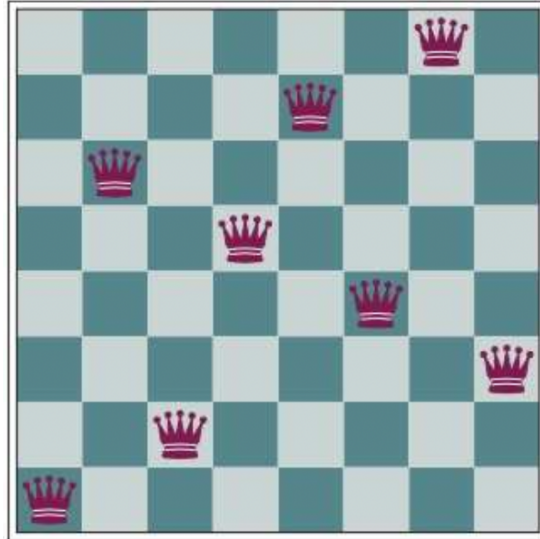Figure 4: 8 Queen board.

Video Recording Link: https://uwin365-my.sharepoint.com/:v:/g/personal/luthraj_uwindsor_ca/ERSvu5cdRJhKsdfyCrJTtPcBJ5C63D6M-GJ8cpARoduciQ?email=danwu%40uwindsor.ca&e=N3OS7A

Link for the Code: https://github.com/jksingh07/AI-Assignment

6 Question 6

Consider the erratic vacuum world first introduced in Ch.4.3.1, if the initial state is state 5 (refer to Fig.4.9 for state numbering), draw the AND-OR search tree similar to the one in Fig. 4.10.

Answer

Search tree for the erratic vacuum world.

State nodes are OR nodes where a decision must be made about what to do. At the AND nodes, shown as circles, every outcome must be handled, as indicated by the arc linking the outgoing branches. Bold lines indicate where the solution was located.

Where, V denotes Vacuum
D denotes Dirt
Methods: left, right, suck

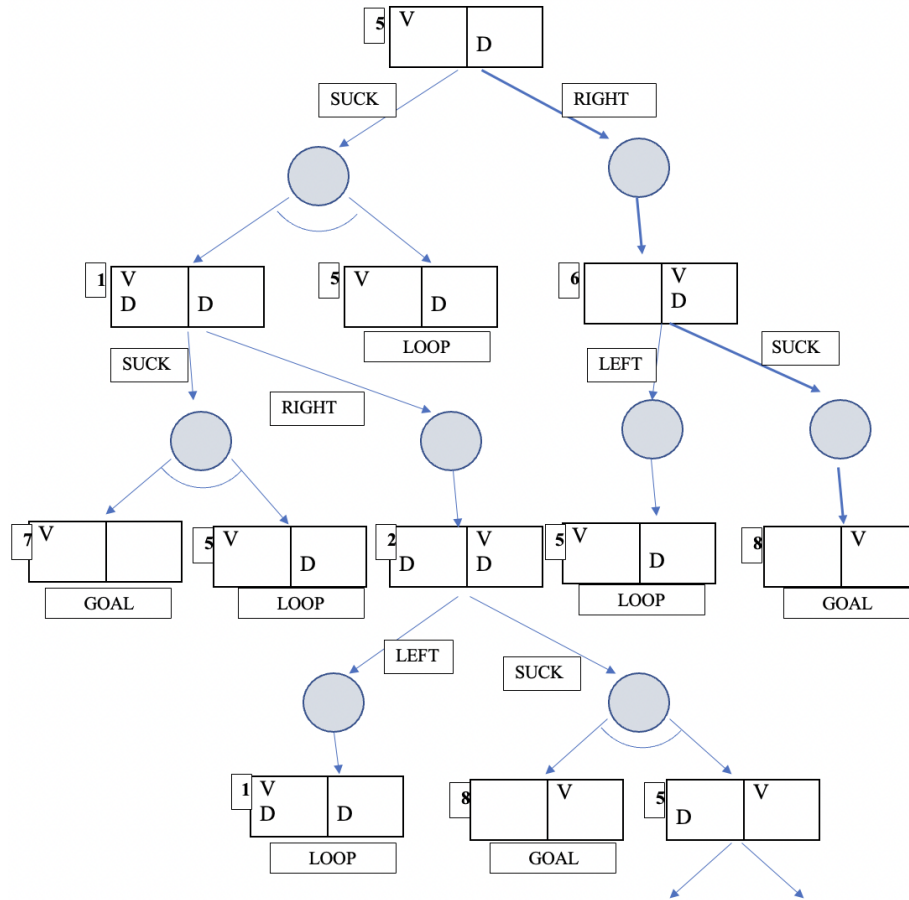

Figure 5: AND-OR Search Tree for erratic vacuum world.

7   Question 7

Consider the slippery vacuum world first introduced in Ch.4.3.3, continue drawing the AND-OR search tree in Fig.4.12 so that we could see at least one goal state is reached.

Answer
In the search graph for a slippery vacuum world, I have explicitly shown (some) cycles. Because there is no reliable way to move, all solutions to this problem are cyclic plans.
V denotes Vacuum
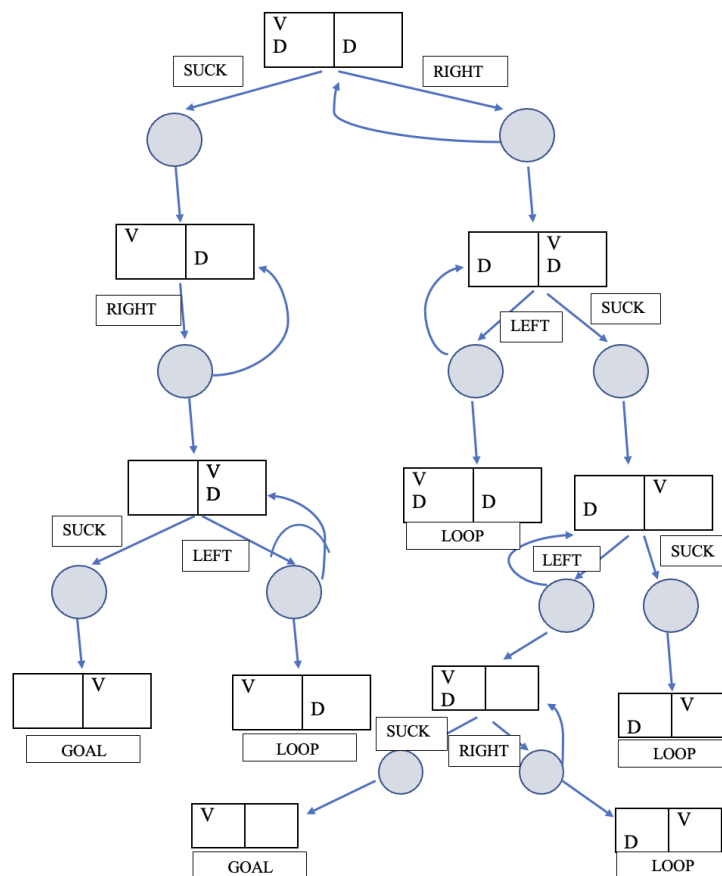**D denotes Dirt**
Methods: left, right, suck

Figure 6:   AND-OR Search Tree for Slippery erratic vacuum world.

References

[RN22] Stuart J. Russell and Peter Norvig. Artificial intelligence: A modern
approach. pages 141–146, 2022.