LAB-3

REPORT

CS-571: ARTIFICIAL INTELLIGENCE

SUBMITTED BY:
ABHIJEET KUMAR (2101CS02)
NAITIK RAJ (2101CS49)
SATYAM KUMAR (2101CS72)

Que 1.

Target State = [[3,2,1], [4, 5, 6], [8,7, 0]]
Given the initial state = [[1,2,3], [4,5,6], [7,8,0]], number of states explored corresponding to different heuristics:

In terms of efficiency, h4 < h1 < h2 < h3 < h5
Now, comparing the number of states expanded by the different heuristics, we can observe:
h4 > h1 > h2 > h3 > h5

Uniform Cost Search (h1) returns 0 for all states, and, thus expands to a larger number of states.
Number of displaced tiles (h2) only calculates the number of tiles displaced from their position in the goal state. It doesn't provide much information about the distance between the current state and the target state.

Manhattan distance (h3) calculates sum of distances (along - row and along - column). 40 (constant value; Inadmissible heuristic) (h4) returns a constant value as the shortest path cost from the current state to the final state. The heuristic doesn't provide any extra information about the current state.

Manhattan Distance + Adjacent Swap Cost (h5) takes into account both the distance of tiles from their goal positions (Manhattan distance) and also adds in +2 for every adjacent pair which are in each other's terminal position.
Thus, h4 performs the worst, h5 provides the best result, followed by h3, which provides a more informed estimate of remaining cost to reach the goal state than h2 and h1.
And accordingly, the number of states explored by h4 is the most, followed by h1, h2, h3 and then h5.

Que 2.

Target State = [[1, 2, 3], [4, 5, 6], [7, 8, 0]] Given the initial state = [[1, 2, 3], [4, 5, 6], [0, 7, 8]] Heuristic h1 generates the following states (open list):

States expanded: 6

Expanded states :

```
123 123 123    123  123  123
456 056 450    456  406  456
078 478 678    708  758  780
```

Heuristic h2 expands the following states:

States expanded: 3

```
123 123 123 456
456 456 078 708
780
```

Heuristic h3 expands the following states:

States expanded: 3

```
123 123 123 456
456 456 078 708
780
```

Heuristic h4 expands the following states:

States expanded: 5

```
123 123 123 123 123
456 456 056 506 456
078 708 478 478 780
```

Heuristic h5 expands the following states:

States expanded: 3

```
123 123 123

456 456 456 078
708 780
```

We know, h4 < h1 < h2 < h3 < h5

Also, from the above example, we can observe that states

```
123   123   123
456   456   456
078   708   780
```

appear in the list of expanded states of heuristic h5, h1 and h2, h3, h4 as well.

Que 3.

<u>Monotone restriction</u>: h(n) <= cost(n,m) + h(m)

a. **<u>h(1) = 0</u>**

   Let us consider 2 states from the previous example. n = { {1, 2, 3}, {4, 5, 6}, {0, 7, 8} }; h(n) = 0 m = { {1, 2, 3}, {0 5, 6}, {4, 7, 8} }; h(m) = 0 such that c(n, m) = 1

   Substituting the values in the equation, we get:

   0 <= 1 + 0

   ⇒ 0 <= 1, which is true

   Hence, proved.

b. **<u>h(2) = Number of displaced tiles</u>**

   Consider two states, n and m such that, n and m are neighbouring states, or c(n, m) = 1. Let, h2(n) = h.

   Now, going from n to m, we move one tile, which can result in a state with:

   - 1 more misplaced tile; h2(m) = h + 1

     Substituting values, h <= 1 + h + 1

     ⇒ h <= h + 2, which is true

   - 1 less misplaced tile; h2(m) = h - 1

     Substituting values, h <= 1 + h - 1

     ⇒ h <= h, which is true

   - same number of misplaced tiles; h2(m) = h

     Substituting values, h <= 1 + h

     ⇒ h <= h + 1, which is true

   <u>Example case 1:</u>

   Let us consider the states: n = { {1, 2, 3}, {4, 5, 6}, {0, 7, 8} }; h(n) = 2 m = { {1, 2, 3}, {0, 5, 6}, {4, 7, 8} }; h(m) = 3 where c(n, m) = 1

   Substituting the values in the equation, we get:

   2 <= 1 + 3

   ⇒ 2 <= 4, which is true

   Hence, proved.

   <u>Example case 2:</u>

Let us consider the same states:
n = { {1, 2, 3}, {4, 5, 6}, {0, 7, 8} }; h(n) = 2

m = { {1, 2, 3} {4, 5, 6}, {7, 0, 8} }; h(m) = 1
where c(n, m) = 1

Substituting the values in the equation, we get:
2 <= 1 + 1
⇒ 2 <= 2, which is true
Hence, proved.

c. **<u>h(3) = Manhattan distance</u>**

Consider two states, n and m such that, n and m are neighbouring states, or c(n, m) = 1. Let, h2(n) = h.
Now, going from n to m, we move one tile, which can result in a state which is:

- away from the goal state; h3(m) = h + 1

  Substituting values, h
  <= 1 + h + 1
  ⇒ h <= h + 2, which is true
- towards the goal state; h3(m) = h - 1

  Substituting values, h
  <= 1 + h - 1
  ⇒ h <= h, which is true

<u>Example case 1:</u>

Let us consider the same states:
n = { {1, 2, 3}, {4, 5, 6}, {0, 7, 8} }; h(n) = 1 + 1 = 2 m
= { {1, 2, 3}, {0 5, 6}, {4, 7, 8} }; h(m) = 1 + 1 + 1 = 3
where c(n, m) = 1

Substituting the values in the equation, we get:
2 <= 1 + 3
⇒ 2 <= 4, which is true
Hence, proved.

<u>Example case 2:</u>

Let us consider the same states: n = { {1, 2, 3},
{4, 5, 6}, {0, 7, 8} }; h(n) = 1 + 1 = 2 m = { {1, 2,
3} {4, 5, 6}, {7, 0, 8} }; h(m) = 1 where c(n, m) =
1

Substituting the values in the equation, we get:
2 <= 1 + 1

⇒ 2 <= 2, which is true
Hence, proved.

d. **<u>h4(n) = 40 (Inadmissible heuristic)</u>**

This heuristic is not admissible because from excessive experimentation, we found out that from any initial state to any terminal state, it takes about 30-odd moves in the optimal path. Thus h4(n) > h*(n).

However this heuristic is monotonic, because it follows the following inequation:

- h4(m) <= h4(n) + 1 (because 40 <= 41)
  Here, m is a state reached from n after making one legal move.

e. **<u>h5(n) = Manhattan Distance + Adjacent Swap Cost</u>**

Consider two states, n and m such that, n and m are neighbouring states, or c(n, m) = 1. Let, h5(n) = h.
Lets define **Special Pair(a, b)** as **a** and **b** being two elements which are in each other's TERMINAL position and adjacent in the present state.
Now, going from n to m, we move one tile, which can result in a state which is:

- If no **Special Pair** is being neither removed nor added:
- away from the goal state; h5(m) = h + 1

  Substituting values, h
  <= 1 + h + 1
  ⇒ h <= h + 2, which is true
- towards the goal state; h5(m) = h - 1

  Substituting values, h <= 1 +
  h - 1 ⇒ h <= h, which is
  true
- If a **Special Pair** is getting formed:
- away from the goal state; h5(m) = h + 1 + 2
  Substituting values,
  h <= 1 + h + 1 + 2
  ⇒ h <= h + 4, which is true -
  towards the goal state; h5(m) = h - 1 + 2

  Substituting values,
  h <= 1 + h - 1 + 2 ⇒ h <=
  h + 2, which is true
- If an existing **Special Pair** is being removed:
- away from the goal state: h5(m) = h + 1 - 2

  Substituting values,
  h <= 1 + h + 1 - 2

⇒ h <= h, which is true

- Can never lead to move towards TERMINAL position and be a **Special pair** at the same time in ONE move.

- A **Special Pair** CANNOT be formed and removed in ONE move.

Que 4.

Considering,
Target State = [[1, 2, 3], [4, 5, 6], [7, 8, 0]]
Given the initial state = [[1, 3, 6], [0, 4, 7], [5, 8, 2]]

Heuristic Function: h(S) = Uniform Cost Search Could
NOT find goal state, visited 181440 grid states.
Time taken for execution: 1941ms

Heuristic Function: h(S) = Displaced tiles
Could NOT find goal state, visited 181440 grid states. Time
taken for execution: 1706ms

Heuristic Function: h(S) = Manhattan distance Could
NOT find goal state, visited 181440 grid states. Time
taken for execution: 2988ms

Heuristic Function: h(S) = 40 (Inadmissible heuristic) Could
NOT find goal state, visited 181440 grid states.
Time taken for execution: 1777ms

Heuristic Function: h(S) = Manhattan Distance + Adjacent Swap Cost
Could NOT find goal state, visited 181440 grid states. Time taken for
execution: 2447ms
The 8-puzzle problem has finite number of states and a valid sequence of moves can lead from initial state to the target state. However, not all target states are reachable from every initial state. Some states maybe isolated from the rest of the states.

In 8-puzzle problem, only half states are reachable from any given state. As observed from the above results, the algorithm has explored all the possible reachable states (9! / 2 = 181440). Now, there is no new path left to explore.
Thus, it suggests that the target state lies in the set of disconnected states, and therefore is unreachable.

Que 6.

Considering heuristic h3(n) = Manhattan Distance, Example case:

Let us consider the states:

Calculating h3(n) considering Manhattan distance of the blank space as well: n = { {1, 2, 3}, {4, 5, 6}, {0, 7, 8} }; h(n) = 1 (d(7))+ 1(d(8)) + 2(d(0)) = 4 m = { {1, 2, 3}, {4, 5, 6}, {7, 0, 8} }; h(m) = 1(d(8)) + 1(d(0)) = 2

where c(n, m) = 1

//where, d(i) denoted Manhattan distance of tile i, w.r.t to the goal state

Substituting the values in the equation, we get:

4 <= 1 + 2

⇒ 4 <= 3, which is false

Hence, proved.

→

The time and space complexities of the A* algorithm depends on how effective the chosen heuristic is. In cases where the search space is unbounded, the worst-case scenario can cause an exponential increase in the number of expanded nodes (which, inturns depends on b, the branching factor). The efficiency of A* search varies according to how much additional information is provided by the heuristic function, as it enables the algorithm to eliminate numerous unexplored states that an uninformed search would need to examine.

As observed in Question 1, the number of explored states for the same randomly generated inital grid and a fixed target state varies grealy across different heuristics.

In the A* algorithm, each iteration involves selecting a node that minimizes the function: f(n) = g(n) + h(n)

Here, g(n) represents the optimal cost from the starting state to the current state 'n,' and h(n) signifies the heuristic function estimating the cheapest path cost from current state 'n' to the goal state.

Efficiency:

Consider two versions of A* with distinct admissible heuristic functions:

f1(n)=g1(n)+h1(n) f2(n)=g1(n)+h2(n)

In our case, the backward cost for all the heuristics remains the same. The difference is made by the the heurisitic.

A* with evaluation function f1 proves to be more informed than A* with f2 when, for all non-goal states, h1(n)>h2(n).

Optimality:

The A* algorithm with **admissible** heuristics, like, Manhattan distance, Number of displaced tiles, guarentees optimality. They optimality guarentees that the path found is the shortest possible path.

Completeness:
The A* algorithm is complete, irrespective of the choice of the heurestics. A* algorithm finds the solution if it exists.