CMPT 310 Assignment 1 Report

Junchen Li

301385486

2021/2/7

1. Experiment Introduction:

We random initial twelve 8 puzzles question and using three different algorithms (A* searching, Manhattan Distance Heuristic, Gashing's Heuristic) to run them on the same computer. I made a table to collect these data and some analyzes.

2. Sample initialization:

1) [2,4,5,1,6,8,0,3,7]
2) [3,0,2,7,4,8,5,1,6]
3) [3,8,1,4,6,5,2,7,0]
4) [2,4,1,5,8,7,6,0,3]
5) [0,1,3,4,2,5,7,8,6]
6) [6,1,0,4,5,2,3,8,7]
7) [8,2,3,0,7,1,5,4,6]
8) [8,0,2,1,4,5,6,7,3]
9) [2,0,1,6,7,3,4,8,5]
10) [5,7,2,8,1,0,3,4,6]
11) [2,8,0,1,7,3,5,4,6]
12) [0,8,3,5,2,1,7,6,4]

3. Data collection

	A*	Manhattan Distance	Gaschnig's Heuristic
1	62.94691586494446	0.328524112701416	23.523566961288452
2	56.56760025024414	0.7015302181243896	34.511847257614136
3	8.206449747085571	0.12789607048034668	2.806095838546753
4	6.115472078323364	0.05018496513366699	3.281949996948242
5	0.000309944152832	0.000252246856689	0.0001800060272216
6	47.64756822586056	0.14039015769958496	15.700300216674805
7	0.932411193847656	0.0177421569824218	0.3278810977935791
8	1.254073858261108	0.0200231075286865	0.715054988861084

9	5.62860894203186	0.10897994041442871	2.0794589519500732
10	4.452006816864014	0.0089910030364990	1.620139837265014
11	0.067202091217041	0.00353097915649414	0.0335280895233154
12	59.08315491676330	0.7948389053344727	30.504348039627075
min	0.000309944152832	0.000252246856689	0.0001800060272216
max	62.94691586494446	0.7948389053344727	34.511847257614136
mean	19,18435715723131	0.15086894678765433	8.0592318467823443

For calculating the mean, we excluded the max value and min value and the rest number of items are ten.

From above, comparisons of running time are A* < Gasching < Manhattan. The Manhattan Distance Heuristic have the best performance. However, from the vertical perspective, the running time is really depending on the difficult level of eight puzzle. More difficult question will cost longer running time

Here, is the screenshot of the code running part.

```
Terminal:  Local  +
(venv) (base) macdeMBP-3:as1 mac$ python3 main.py 1 3 4 8 6 2 0 7 5
[1, 3, 4, 8, 6, 2, 0, 7, 5]
 The original is not solvability
[2, 5, 0, 1, 6, 4, 8, 7, 3]
 Unsolved state :
2  5  *
1  6  4
8  7  3
----->  A*-search by using the !! misplaced tile heuristic !!  <-----
The total running time in A* searching algorithm is (in seconds): 64.73416805267334
The total nodes that were expanded are 13441
The total length of moving for the solution are 24
----->  A*-search by using the !! Manhattan Distance Heuristic !!  <-----
The total running time in manhattan Distance algorithm is (in seconds): 0.8712940216064453
The total nodes that were expanded are 1590
The total length of moving for the solution are 24
----->  A*-search by using the !! Gaschnig_Heuristic !!  <-----
The total running time in gaschnig Heuristic algorithm is (in seconds): 22.494899034500122
The total nodes that were expanded are (<Node (1, 2, 3, 4, 5, 6, 7, 8, 0)>, 7883)
(venv) (base) macdeMBP-3:as1 mac$
```

4. Discussion:

1. Based on your data, can you give a recommendation for what you think is the best algorithm? Explain how you came to your conclusion.

Answer: Base on some searching information, in some notes says the Gaschnig will the best one and it has a nick name called "nine steps algorithm". But why my algorithm is not working in the experiment, my guess is in the limitation condition the Manhattan will have better performance. Also the differences of goal state will also cause the different performance. In my opinion, the big o of last two them are same, Manhattan solves the problem better mathematically, not because the third one has no restrictions and can move freely.

2. Is it possible to rule out one of the heuristic functions without looking at experimental data, but purely based on theoretical consideration? Hint: We say that an admissible heuristic h_1 is at least as accurate as another admissible heuristic h_2 iff for every n, $h_1(n) \geq h_2(n)$.

Answer: If we didn't look at the data result, the Manhattan algorithm could be work. Because it is based on mathematically knowledge.

3. Can you suggest a heuristic that is always at least as accurate as all 3 heuristic functions discussed here?

Answer: In my opinion, the most accurate way is just show all possible outcome. The algorithm that I think out is the Brute-force. We can perform a breadth-first search on the state space tree and it always finds the target state which closest to the root. I guess this one can did as accurate as the all 3 heuristic functions