CSC 425/525 Artificial Intelligence
Search Algorithms for Eight-puzzle Game

Semester: Fall 2020

Instructor: Dr. Junxiu Zhou

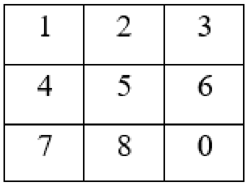
Introduction

The eight-puzzle game is a 3 × 3 version of the 15-puzzle in which eight tiles can be moved around in nine spaces (an amusing game can be played here 8-puzzle (https://murhafsousli.github.io/8puzzle/#/)). Because the eight-puzzle game generates a smaller state space than the full 15-puzzle and its graph fits easily on a page, however they follow the same state space representation and search strategies, therefore we use eight puzzle game as our project in this course.

There are eight numerical tiles and one blank tile in the eight-puzzle game. We specify a beginning state and a goal state for the 8-puzzle, it is possible to give a state space accounting of the problem-solving process, as shown below two figures for the example of beginning and goal state of the eight-puzzle game.

2	3	6
1	4	8
7	5	0

Fig. 1 An example of the beginning state of the eight-puzzle game (here 0 stands for the blank tile)



1	2	3
4	5	6
7	8	0

Fig. 2 The goal state of the eight-puzzle game (here 0 stands for the blank tile)

A state is a panel contains the 9 spaces with 9 tile numbers. The blank tile is moved legally in the 3 × 3 panel until the state reaches the final goal state. Fig. 3 exemplifies partial of a state generation graph with state space and arcs. A naive implementation of brute force exhaustive search may successfully find the goal state. But the problem size increases as the puzzle size increases.

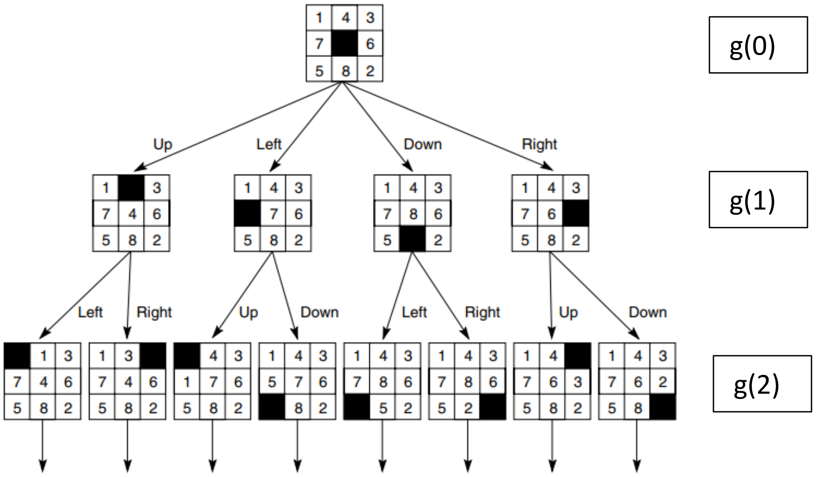


Fig. 3 A partial example of a state generation graph of the eight-puzzle game.

The state space of 8-puzzle is represented by a rooted graph such as Fig. 3. The legal moves are:

a) move the blank up ↑
b) move the blank right →
c) move the blank down ↓
d) move the blank left ←

Each time, the game allows one tile replace. In the provided code implementation, tile labeled with 0 represents the blank tile. The difference between start state and the goal state is the crux for solving the game.

Project Requirement

In this project, we need to design and implement different search algorithms that we learned in the AI class to solve the eight-puzzle game problem automatically. The detailed requirements are:

1. Using uninformed search method to solve the 8-puzzle game problem (30%)

Please select one of the learned uninformed search algorithms (breadth-first search, depth-first search, or iterative deepening search) to implement to solve the 8-puzzle game problem. More details about these three algorithms can be found in Chapter 3.2.3 and 3.2.4.

2. Informed search (50%)

Please use the best_first_search algorithm along with heuristic search rules to construct the body of the program to solve the 8-puzzle game problem. Similar to uninformed search method, best-first search uses lists to maintain states:

- an OPEN list is used to store the unexplored states
- a CLOSE list is used to store the visited state

OPEN list is a priority queue. The priority is insured through sorting the OPEN list each time after new states are generated and added into the list. At each iteration, best_first_search removes the first element from the OPEN list. If it meets the goal conditions, the algorithm returns the solution path that led to the goal. Note that each state retains ancestor information to determine if it had previously been reached by a shorter path and to allow the algorithm to return the final solution path.

The heuristics are used as sorting criteria. In this informed search, reducing the state space search complexity is the main criterion. We define heuristic evaluations to reduce the states that need to be checked every iteration. Evaluation function is used to express the quality of informedness of a heuristic algorithm.

Evaluation function $f(n)$

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

$g(n)$ measures the actual length of the path from any state n to the start state
$h(n)$ is a heuristic estimate of the distance from state n to a goal.

In this 8-puzzle game project, we have

$g(n)$ = actual distance from n to the start state
$h(n)$ = Number of tiles out of place + Sum of distance out of place + Number of direct tile of reversals

As can be see that three heuristics are adopted in this implementation:

A. Number of tiles out of place
Count the number of tiles that are misplaced through compare the start and goal state.

B. Sum of distance out of place
Count the number of steps needed to move the tile to the correct location. If we treats the 8-puzzle as a 2d array, then this part can be treated as the absolute value of offsets between the correct location in goal state and deviant location in start state of the same tile.

C. Number of direct tile of reversals
Number of the adjacent tile pair which the location are reversed.

3. Comparison analysis. (10%)

Please test the search process of your implemented uninformed search algorithm and informed search algorithm in terms of computational cost and the length of the path for solving the 8-puzzle game problem. Write down your detailed analysis and comparison conclusion.

Note that,

(1) to get the computational time of specific program statements or segments you can use the following to statements

```
t0 = time.time()
your program statements
t1 = time.time()
ComputationalTime = (t1-t0)
```

(2) to get fair comparison, you better to use the same experimental platform (the same computer)

4. Bonus points (Optional) (10%)

Employing more informed heuristic estimation. By adding more restrictions into the heuristic test function. Currently there are three heuristics which elaborated in the informed search section. There may exist other possible heuristics, for example, recursively check the shortest path, fine tune the parameter, and so on. Especially, for complicated initial state in some case, it took a long time to reach the goal by using the informed search algorithm mentioned above. If your newly added heuristic rule(s) improved the search process compared with the above uninformed and informed search algorithms, then you will get this bonus points.

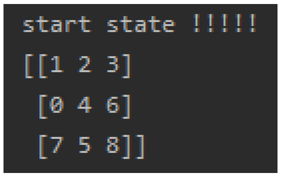
5. Project report (10%)

You have to write a project report for this project. You have to complete the template and submit it together with your project source code. The length of the report must longer than 1 page. The template of the project report is attached in the project zip file.

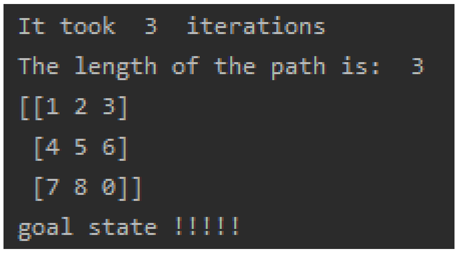
Result and Discussion

The output is the path from the original state to the goal state. We can use this to verify the correctness and optimality of our algorithms. Two sample runs are shown below (of course, you can define the display mode by yourself). Note that these two sample runs, shows the results of Informed Search Algorithm only:

Sample Run 1:

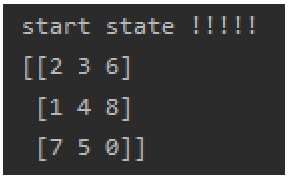


```
start state !!!!!
[[1 2 3]
 [0 4 6]
 [7 5 8]]
```

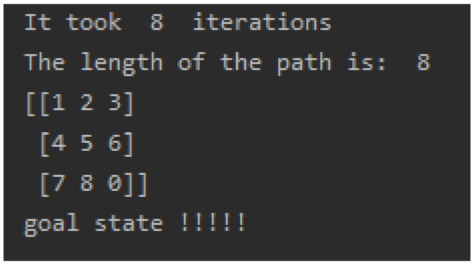


```
It took  3  iterations
The length of the path is:  3
[[1 2 3]
 [4 5 6]
 [7 8 0]]
goal state !!!!!
```

Sample Run 2:



```
start state !!!!!
[[2 3 6]
 [1 4 8]
 [7 5 0]]
```



```
It took  8  iterations
The length of the path is:  8
[[1 2 3]
 [4 5 6]
 [7 8 0]]
goal state !!!!!
```

Hand In Requirements

(1) The starter code is provided in Python language. You must follow the provided starter code to finish this project. There are two reasons for this:

   a. To prevent plagiarism;
   b. Reading and understanding other people's code should be one the ability of CS students.

   Note that if your team choose to use Java to implement this project, please contact me for the information about the starter code.

(2) If you completed this project, you need to hand in the **source code** been completed by you or your group. Your source code compressed to a single ZIP file. The name of the code archive should be **EightPuzzle_Teamname.zip.**
   The code should be well commented, and it should be easy to see the correspondence between what's in the code and what's in the report. You don't need to include executable or various supporting files (e.g., utility libraries) whose content is irrelevant to the assignment. If the

instructor finds it necessary to run your code in order to evaluate your solution, the instructor will get in touch with you.

(3) You need to run both your solvers on several input puzzles and included the results in your project report.
Your output when run the below two initial state:

```
2 3 6     2 3 1          the goal state is 1 2 3
1 4 8     0 4 6                            4 5 6
7 5 0     7 5 8                            7 8 0
```

(4) A project report in PDF format

- The report should briefly describe your implemented solution and fully answer all the questions posed above. **Remember**: *you will not get credit for any solutions you have obtained, but not included in the report*.
- All group reports need to include paragraph of individual contribution, i.e., which group member was responsible for which parts of the solution and submitted material. The instructor reserves the right to contact group members individually to verify this information.
- Describe the solution and compare the number of attempted search steps and execution time for your uninformed search method and informed search method implementations.
- The name of the report file should be **EightPuzzle_Teamname.pdf**. Don't forget to include the names of all group members and the number of credit units at the top of the report. (*You can find the report template on the provided zip file together with the starter code.*)
- Finally, which part you think can be improved and if you have better heuristic estimation. Or the challenges you encountered during this project.

Note that, the instructor reserves the right to give **bonus points** for any advanced exploration or especially challenging or creative solutions that you implement. **If you submit any work for bonus points, be sure it is clearly indicated in your report.**

WARNING: You will not get credit for any solutions that you have obtained, but not included in your report!