Assignment 1 - Using Informed and Uninformed Search Algorithms to Solve 8-Puzzle
Omar Mohamed AbdelBaset Mohamed
44

Heuristics:

- The Manhatan Distance Heuristic is not admissible

Consider state

3	1	2
6	4	5
	7	8

3	1	2
	4	5
6	7	8

	1	2
3	4	5
6	7	8

The total cost = 2 moves

The manhatan distance = 2 + 0 + 0 + 1 + 0 + 0 + 2 + 0 + 0 + 0

- => Over estimates a solution => not admissible
- The Eucledian Distance Heuristic is more admissible than the Manhatan distance as the result is always smaller or equal to the other's.

Example where the Manhatan distance fails:

5	7	0
6	8	1
3	2	4

 A^* with manhatan => cost = 24

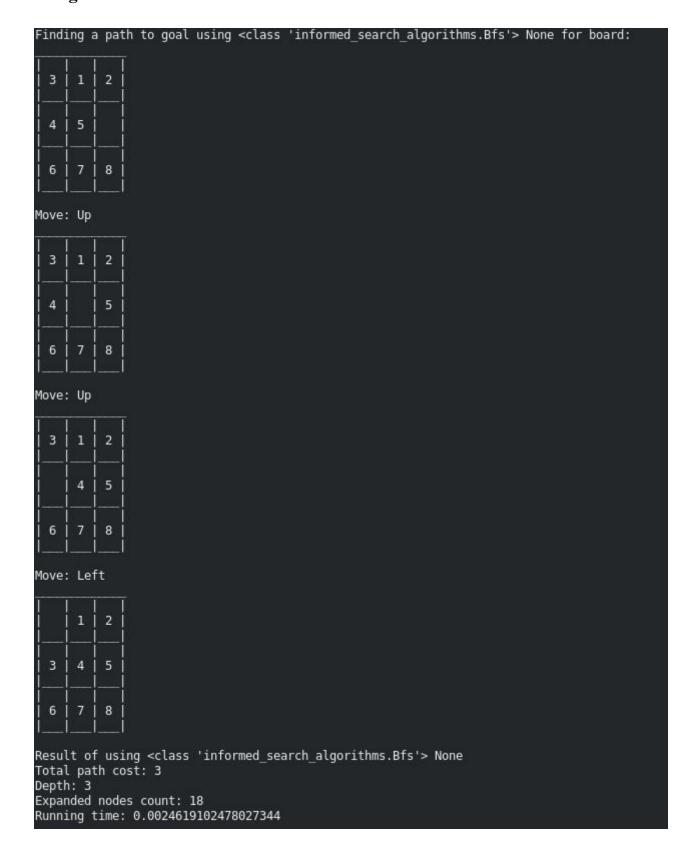
A* with Eucledian \Rightarrow cost = 22

Used Data Structures:

- PuzzleBoard: a 2d tupple representing the puzzle.
- GameState(PuzzleBoard, move, cost, depth): Holds information about the current game state,
 passing this to the searching algorithms along with the callback needed to evaluate
 Neighbouring states from current state.
- Queue: Used in the BFS searching algorithm
- Priority Queue: Used in the A* searching algorithm, using the heuristic function(state) + state.cost to compare between the states in the queue.
- GameStateWrapper: a wrapper class around the GameState class to add functionality to evaluate the selected heuristic function upon comparing it to another state. Used in the priority queue in the A* algorithm.

Samples Runs

Using BFS:



								8 3	
Finding a	a path	to goal	using <cl< td=""><td>ass 'inform</td><td>ed_search_</td><td>algorithms</td><td>.Bfs'> No</td><td>one for</td><td>board:</td></cl<>	ass 'inform	ed_search_	algorithms	.Bfs'> No	one for	board:
3 1	2								
6 4	 5 								
 7 8 									
Move: Up									
3 1									
6 4	5								
7	8								
Move: Up	''								
3 1									
6 4	5								
 7 -									
Move: Le	ft								
3 1	 2								
4	 5 								
6 7	8								
Move: Le	ft								
	2								
3 4	5								
6 7	 8								
Result o	rl f using	<class< td=""><td>'informed</td><td>_search_algo</td><td>orithms.Bf</td><td>s'> None</td><td></td><td></td><td></td></class<>	'informed	_search_algo	orithms.Bf	s'> None			
Total par Depth: 4 Expanded	nodes	count: 3	30	0.00					
Running t	time: 0	+0 7001	171752929		nd coarch	alaarithma	Difets No	ona for	boarde

Finding a path to goal using <class 'informed_search_algorithms.bfs'=""> None for board:</class>
_
7 4 8
Move: Right
3 1 2
6 5
7 4 8
Move: Right
Move: Up
6 4 5 <u></u>
7 8
Move: Left
Move: Left
3 4 5
Result of using <class 'informed_search_algorithms.bfs'=""> None Total path cost: 5 Depth: 5</class>
Expanded nodes count: 34 Running time: 0.0014576911926269531

Using DFS:

It was mostly stack overflows as the path was so deep and python doesn't have many optimization to recursion like tail recursion.

```
Finding a path to goal using <class 'informed_search_algorithms.Dfs'> None for board:
     4
         R
Fraceback (most recent call last):
 File "index.py", line 96, in <module>
 evaluate(board, Dfs)
File "index.py", line 50, in evaluate
init_state, puzzle.GOAL_STATE, puzzle.generate_neighbours)
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/informed_search_algorithms.py", line 74, in solve
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/informed_search_algorithms.py", line 62, in _solve
   parents, expanded_nodes_cnt)
  File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 62, in solve
   parents, expanded_nodes_cnt)
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/informed_search_algorithms.py", line 62, in _solve
   parents, expanded_nodes_cnt)
  [Previous line repeated 985 more times]
  File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 59, in solve
   for neighbour in reversed(get_neighbours_fn(current_state)):
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 119, in generate_neighbours
   move=move[2]) for move in _generate_valid_moves(state.board)
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 119, in <listcomp>
   move=move[2]) for move in generate valid moves(state.board)
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 105, in _transition

 for row in range(N)

  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 105, in <listcomp>
    ]) for row in range(N)
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 104, in <listcomp>
    for col in range(N)
  File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 89, in _transition_get_item
   if (row, col) == empty_space_pos:
RecursionError: maximum recursion depth exceeded in comparison
```

Finding a path to goal using <class 'informed search algorithms.Dfs'> None for board:

```
3
          2
          5
6
     4
7
     8
```

```
Traceback (most recent call last):
```

File "index.py", line 96, in <module> evaluate(board, Dfs)

File "index.py", line 50, in evaluate
 init_state, puzzle.GOAL_STATE, puzzle.generate_neighbours)

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 74, in solve set(), [], 0)

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 62, in solve parents, expanded nodes cnt)

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 62, in solve parents, expanded nodes cnt)

File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/informed_search_algorithms.py", line 62, in _solve parents, expanded_nodes_cnt)
[Previous line repeated 985 more times]

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 59, in solve for neighbour in reversed(get_neighbours_fn(current_state)):

File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 119, in generate_neighbours move=move[2]) for move in generate valid moves(state.board)

File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 119, in <listcomp> move=move[2]) for move in generate valid moves(state.board)

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/puzzle.py", line 105, in transition]) for row in range(N)

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/puzzle.py", line 105, in stcomp>]) for row in range(N)

File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 104, in listcomp> for col in range(N)

File "/home/vontman/Dropbox/projects/8 Puzzle Ai/puzzle.py", line 89, in transition get item if (row, col) == empty space pos:

RecursionError: maximum recursion depth exceeded in comparison

Finding a path to goal using <class 'informed search algorithms.Dfs'> None for board:

```
3 1 2 1 4 5 1 4 5 1 6 7 8 1
```

```
Traceback (most recent call last):
 File "index.py", line 96, in <module>
  evaluate(board, Dfs)
 File "index.py", line 50, in evaluate
    init_state, puzzle.GOAL_STATE, puzzle.generate_neighbours)
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 74, in solve
    set(), [], 0)
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 62, in solve
   parents, expanded nodes cnt)
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search algorithms.py", line 62, in solve
   parents, expanded_nodes cnt)
 File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/informed_search_algorithms.py", line 62, in _solve
 parents, expanded_nodes_cnt)
[Previous line repeated 985 more times]
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/informed search_algorithms.py", line 59, in _solve
    for neighbour in reversed(get neighbours fn(current state)):
 File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 119, in generate_neighbours
   move=move[2]) for move in _generate_valid_moves(state.board)
 File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 119, in listcomp>
   move=move[2]) for move in _generate_valid_moves(state.board)
 File "/home/vontman/Dropbox/projects/8_Puzzle_Ai/puzzle.py", line 105, in _transition
    ]) for row in range(N)
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/puzzle.py", line 105, in listcomp>
    ]) for row in range(N)
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/puzzle.py", line 104, in listcomp>
    for col in range(N)
 File "/home/vontman/Dropbox/projects/8 Puzzle Ai/puzzle.py", line 89, in transition get item
    if (row, col) == empty_space_pos:
RecursionError: maximum recursion depth exceeded in comparison
```

Using A* with Manhatan Distance:

Finding a path to goal using <class 'uninformed_search_algorithms.astar'=""> <function 0x7f833e13c9d8="" at="" manhatanheuristic=""> for board:</function></class>
4 5
6 7 8
Move: Up
Move: Up
4 5
Move: Left
Result of using <class 'uninformed_search_algorithms.astar'=""> <function 0x7f833e13c9d8="" at="" manhatanheuristic=""> Total path cost: 3 Depth: 3 Expanded nodes count: 4</function></class>
Running time: 0.00431513786315918

Find:	ing a	a pat	h to goal using <class< th=""><th>'uninformed_search_a</th><th>llgorithms.AStar'></th><th><function manha<="" th=""><th>atanHeuristic a</th><th>nt 0x7f3f08e379d</th><th>8> for board:</th></function></th></class<>	'uninformed_search_a	llgorithms.AStar'>	<function manha<="" th=""><th>atanHeuristic a</th><th>nt 0x7f3f08e379d</th><th>8> for board:</th></function>	atanHeuristic a	nt 0x7f3f08e379d	8> for board:
 3 	1	 2 							
 6	4	5							
 7 	 8 								
Move	: Up								
3	1	2							
 6 	 4 	5							
7		 8 							
Move	: Up								
3	1	 2							
6	4	5							
i	 7 	8							
Move	: Le	ft							
3	1	2							
į –	4	5							
6	7	8							
Move	: Le	ft							
	1	2							
3	4	5							
6	7	 8							
Total Depti Expan	l pat h: 4 nded	th co node	ng <class 'uninformed_<br="">st: 4 s count: 5 0.003524303436279297</class>	search_algorithms.ASt	ar'> <function man<="" td=""><td>nhatanHeuristic</td><td>at 0x7f3f08e37</td><td>⁷9d8></td><td></td></function>	nhatanHeuristic	at 0x7f3f08e37	⁷ 9d8>	

rinding a path to goat using <ctass uninformed_search_atgorithms.astar=""> <runction 0x="" a="" at="" bi9uo="" fc0="" mannatanneuristic=""> for board;</runction></ctass>
Move: Right
Move: Right
Move: Up
_ _
111
_ _ Move: Left
T
3 1 2
[
Move: Left
1 3 1 4 1 5 1 !!!
Result of using <class 'uninformed_search_algorithms.astar'=""> <function 0x7fc07a7b19d8="" at="" manhatanheuristic=""> Total path cost: 5 Depth: 5 Expanded nodes count: 14 Running time: 0.014968156814575195</function></class>
Running time: 0.014968156814575195

Using A* with Eucledian Distance:

Findi	.ng a	a path	to goa	l using	<class< th=""><th>'uninfor</th><th>med_sear</th><th>ch_algor</th><th>ithms.A</th><th>Star'></th><th><function< th=""><th>n eucled</th><th>ianHeuri</th><th>stic at</th><th>0x7f833</th><th>e13c950></th><th>for I</th><th>board:</th></function<></th></class<>	'uninfor	med_sear	ch_algor	ithms.A	Star'>	<function< th=""><th>n eucled</th><th>ianHeuri</th><th>stic at</th><th>0x7f833</th><th>e13c950></th><th>for I</th><th>board:</th></function<>	n eucled	ianHeuri	stic at	0x7f833	e13c950>	for I	board:
 3 	1	2																
4	5																	
 6 	7	 8 																
Move:	Up																	
 3 	1	2																
4 		5																
 6 	7	8																
Move:	Up																	
3	1	2																
	4	5																
 6 	7	 8 																
Move:	Le	ft																
 	1	 2 																
 3 	4	 5 																
6 6 	7	 8 																
Total	. pat	th cos	t: 3			earch_al	gorithms	.AStar'>	<funct< td=""><td>ion euc</td><td>ledianHeu</td><td>ıristic</td><td>at 0x7f8</td><td>33e13c9</td><td>50></td><td></td><td></td><td></td></funct<>	ion euc	ledianHeu	ıristic	at 0x7f8	33e13c9	50>			
Expar Runni	ng 1	nodes time:	count: 0.00457	4 7636718	75			,										

Depth: 4	Findi	ing a	path	to goal	lusing	<class< th=""><th>'uninforme</th><th>d_search</th><th>_algori</th><th>thms.ASta</th><th>r'> <func< th=""><th>tion euc</th><th>ledianHeur</th><th>istic</th><th>at 0x7f3</th><th>f08e37950:</th><th>for</th><th>board:</th></func<></th></class<>	'uninforme	d_search	_algori	thms.ASta	r'> <func< th=""><th>tion euc</th><th>ledianHeur</th><th>istic</th><th>at 0x7f3</th><th>f08e37950:</th><th>for</th><th>board:</th></func<>	tion euc	ledianHeur	istic	at 0x7f3	f08e37950:	for	board:
Move: Up 3 1 2	 3 	1	2															
Move: Up 3 1 2 6 4 5 7 8 8 9 9 1 1 1 1 1 1 1 1		4	5															
3 1 2	 7 	8																
	Move:	Up																
	 3 	1	2															
Move: Up	 6 	4	5															
	 7 		8															
	Move:	Up																
	 3 	1	2															
Move: Left	 6 	4	5															
	I I I	7	 8 															
3 1 2	Move:	Let	ft															
	 3 																	
	 6 	7	 8 															
	Move:	Let	ft															
_ _ 6 7 8 _ _ _ _ Result of using <class 'uninformed_search_algorithms.astar'=""> <function 0x7f3f08e37950="" at="" eucledianheuristic=""> Total path cost: 4 Depth: 4</function></class>	I I	1	2															
<pre>I Result of using <class 'uninformed_search_algorithms.astar'=""> <function 0x7f3f08e37950="" at="" eucledianheuristic=""> Total path cost: 4 Depth: 4</function></class></pre>	 3 	4	5															
Depth: 4	 6 	7	 8 															
	Depth	1: 4					earch_algo	rithms.	AStar'> •	function	eucledia	nHeurist	ic at θx7f	3f08e3	7950>			

Findi	ng	а ра	n to goal using <class 'uninformed_search_algorithms.astar'=""> <function 0x7fc07a7b1950="" at="" eucledianheuristic=""> for board:</function></class>
 3		 2 	
 6 	1	i 5 -	
 7 	4	 8 _l	
Move: Right			
 3 	1	 2 	
 6 		 5 _l	
 7 	4	 8 _l	
Move: Right			
 3 	1	 2 _	
 6 	4	 5 _	
 7 		 8 _l	
Move: Up			
 3 	1	 2 	
 6 	4	 5 	
] 	7	 8 _l	
Move: Left			
 3 	1	 2 	
 	4	 5 _	
 6 	7	 8 _l	
Move: Left			
[1	 2 	
 3 	4	[[5 -[
 6 	7	 8 _	
Result of using <class 'uninformed_search_algorithms.astar'=""> <function 0x7fc07a7b1950="" at="" eucledianheuristic=""> Total path cost: 5 Depth: 5 Expanded nodes count: 14 Running time: 0.005842924118041992</function></class>			

Notes:

- Currently to run the code, python3 index.py.
- Added docstrings for the main functions.
- Using mypy with python 3 to add static typing.
- Applying some functional programming concepts so I am sorry for the complication.
- The code could use some refactoring here and there.
- Python is not a great language to write templated and well structured code.