User manual and results

When using h1 (number of tiles out-of-place) as the heuristic function

Let:

```
def heuristic(node):
    return outOfPlace_heur(node)
```

When using h2 (sum of manhattan distance) as the heuristic function

Let:

```
def heuristic(node):
    return manhattanDist_heur(node)
```

The result below is based on h2

1. Esay case

Run this program, solutions to the easy case by six algorithms will be shown as below:

dfs:

goal found

memory needed: 1140

nodes visited: 910

time needed: 0.037897348403930664

path: (truncate the output for too long)

bfs:

goal found

memory needed: 51

nodes visited: 39

time needed: 0.004046201705932617

path: ['UP', 'RIGHT', 'UP', 'LEFT', 'DOWN']

ids:

goal not found

goal not found

goal not found

goal not found

goal not found

goal found

memory needed: 9

nodes visited: 84

time needed: 0.0069732666015625

path: ['UP', 'RIGHT', 'UP', 'LEFT', 'DOWN']

greedy:

goal found

memory needed: 11

nodes visited: 6

time needed: 0.000989675521850586

path: ['UP', 'RIGHT', 'UP', 'LEFT', 'DOWN']

astar:

goal found

memory needed: 14

nodes visited: 11

time needed: 0.0009968280792236328

path: ['UP', 'RIGHT', 'UP', 'LEFT', 'DOWN']

idastar:

goal not found

goal not found

goal not found

goal not found

goal not found

goal found

memory needed: 17

nodes visited: 77

time needed: 0.006978750228881836

path: ['UP', 'RIGHT', 'UP', 'LEFT', 'DOWN']

2. Medium case

Comment on this line:

```
initial_matrix = [[1, 3, 4], [8, 6, 2], [7, 0, 5]]
```

Release the comment on this line:

```
# initial_matrix = [[2, 8, 1], [0, 4, 3], [7, 6, 5]]
```

Run this program, solutions to the medium case by six algorithms will be shown as below:

dfs:

goal found

memory needed: 40339

nodes visited: 32555

time needed: 7.389894247055054

path: (truncate the output for too long)

bfs:

goal found

memory needed: 410

nodes visited: 399

time needed: 0.017951250076293945

path: ['UP', 'RIGHT', 'RIGHT', 'DOWN', 'LEFT', 'LEFT', 'UP', 'RIGHT', 'DOWN']

ids:

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal found

memory needed: 14

nodes visited: 939

time needed: 0.037930965423583984

path: ['UP', 'RIGHT', 'RIGHT', 'DOWN', 'LEFT', 'LEFT', 'UP', 'RIGHT', 'DOWN']

greedy:

goal found

memory needed: 426

nodes visited: 426

time needed: 0.254321813583374

path: ['UP', 'RIGHT', 'RIGHT', 'DOWN', 'LEFT', 'LEFT', 'UP', 'RIGHT', 'DOWN']

astar:

goal found

memory needed: 108

nodes visited: 96

time needed: 0.019945144653320312

path: ['UP', 'RIGHT', 'RIGHT', 'DOWN', 'LEFT', 'LEFT', 'UP', 'RIGHT', 'DOWN']

idastar:

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal found

memory needed: 118

nodes visited: 800

time needed: 0.13264942169189453

path: ['UP', 'RIGHT', 'RIGHT', 'DOWN', 'LEFT', 'LEFT', 'UP', 'RIGHT', 'DOWN']

3. Hard case

Comment on this line:

```
initial_matrix = [[2, 8, 1], [0, 4, 3], [7, 6, 5]]
```

Release the comment on this line:

```
# initial_matrix = [[5, 6, 7], [4, 0, 8], [3, 2, 1]]
```

Run this program, solutions to the hard case by six algorithms will be shown as below:

dfs:

goal found

memory needed: 92588

nodes visited: 76101

time needed: 77.81728434562683

path: (truncate the output for too long)

bfs:

goal found

memory needed: 60964

nodes visited: 181363

time needed: 14.508079290390015

path: ['DOWN', 'RIGHT', 'UP', 'UP', 'LEFT', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'UP', 'LEFT', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'UP', 'LEFT', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'UP', 'LEFT', 'DOWN']

ids:

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal not found

goal found

memory needed: 45

nodes visited: 424473

time needed: 16.705535173416138

path: ['DOWN', 'RIGHT', 'UP', 'UP', 'LEFT', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'UP', 'LEFT', 'DOWN', 'LEFT', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'UP', 'LEFT', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'LEFT', 'UP', 'RIGHT', 'DOWN', 'LEFT']

greedy:

goal found

memory needed: 3693

nodes visited: 4063

time needed: 20.95219349861145

path: ['DOWN', 'RIGHT', 'UP', 'UP', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'UP', 'UP', 'LEFT', 'DOWN', 'DOWN', 'RIGHT', 'UP', 'LEFT', 'LEFT', 'DOWN', 'RIGHT', 'UP', 'LEFT', 'UP', 'RIGHT', 'DOWN', 'DOWN', 'RIGHT', 'UP', 'LEFT', 'LEFT', 'DOWN', 'RIGHT', 'UP', 'RIGHT', 'DOWN', 'LEFT', 'LEFT', 'UP', 'UP', 'RIGHT', 'DOWN', 'RIGHT', 'UP', 'LEFT', 'LEFT', 'DOWN', 'RIGHT', 'RIGHT', 'UP', 'LEFT', 'DOWN']

astar:

Can not solve the hard case in 10 minutes.

idastar:

Can not solve the hard case in 10 minutes.

Analysis

h1: number of tiles out-of-place for heuristic function

h2: sum of manhattan distance for heuristic function

Easy case						
	dfs	bfs	ids	greedy(h1/h2)	astar(h1/h2)	idastar(h1/h2)
memory needed	1140	51	9	12/11	11/14	16/17
nodes visited	910	39	84	6/6	6/11	72/77
time needed	0.038	0.004	0.007	0.0/0.001	0.0/0.001	0.005/0.007
Maximum depth of the recursion						6/6

Medium case						
	dfs	bfs	ids	greedy(h1/h2)	astar(h1/h2)	idastar(h1/h2)
memory needed	40339	410	14	125/426	44/108	121/118
nodes visited	32555	399	939	157/426	33/96	737/800
time needed	7.390	0.018	0.038	0.030/0.254	0.003/0.020	0.103/0.133
Maximum depth of the recursion						10/10

Hard case						
	dfs	bfs	ids	greedy(h1/h2)	astar(h1/h2)	idastar(h1/h2)
memory needed	92588	60964	45	1478/3693	Can not solve the hard case in 10 minutes	
nodes visited	76101	181363	424473	1799/4063		
time needed	77.817	14.508	16.706	3.148/20.952		
Maximum depth of the recursion						

1. For easy case:
   Ids needs the least memory, but greedy and astar(h1) spend the least time.
   For medium case:
   Ids needs the least memory, but astar(h1) spends the least time.
   For hard case:
   Ids needs the least memory, but bfs spends the least time.

2. H1 makes the algorithm spend less space and time for running the program than H2.

3. For dfs:
   Need the most memory and time, can not get the best solution.
   For bfs:
   Need much memory and time, but can get the best solution.
   For ids:
   Need the least memory, but need much time and can not get the best solution.
   For greedy:
   Need much memory and time and can not get the best solution.
   For astar:
   Memory and time spent are little, but when the problem is complex, it can not solve the problem within the short time.
   For idastar:
   In these three cases, cannot perform better than astar.

4. The reason of why astar and idastar failed may be the heuristic function is not so reasonable.