- Due Tuesday by 11:59pm
- Points 10
- Submitting a file upload
- Available Oct 23, 2011 at 12am - Nov 3 at 11:59pm about 9 years

1. (7 points) You will be implementing breadth first search to solve a sliding tile puzzle called 8-puzzle (smaller version of 15-puzzle).

8-puzzle has 8 square tiles, numbered from 1 to 8, arranged in a 3 X 3 square area with one of the 9 cells left vacant. Any tile which is adjacent to the empty cell may be moved into the empty cell. The problem is to take one configuration of the puzzle -- the initial state -- to another configuration -- the goal state -- with a sequence of moves. There is an underlying ``state space graph," where each vertex corresponds to a configuration of the puzzle and there is an edge from u to v if there is a move that can take the configuration from u to v. Each edge also has a label associated with it - D for down move, U for up move, L for left and R for right. The sequence of labels in the path from the initial state to the goal state is called a solution.

For example, the following problem can be solved in 8 moves: D, R, R, U, L, L, D, R.

```
start       goal
1 2 3        1 2 3
4 5 6  ==>   8   4
8 7          7 6 5
```

The state space graph consists of two connected components. Fix the goal for all problems to be the same as above.

- Your program uses breadth first search to solve all problems by searching backwards from the goal (and remembering all states visited). You might use hashing to check if a node has been already visited.
- At each depth, report the number of nodes searched and the cumulative time taken to complete the search to that depth.
- Also report the hardest problem (whose solution is the longest) solved and its solution.

2. (1 point) Suppose that the only negative edges are those that leave the starting node s. Does Dijkstra's algorithm find the shortest path from s to every other node in this case? Justify your answer carefully.

3. (1 point) Give an O(n^3) algorithm that takes a directed graph as input and returns the length of the shortest cycle in the graph where n is the number of nodes.

4. (1 point) You are given a strongly connected directed graph G = (V,E) with positive edge weights. Give an efficient algorithm for finding the shortest paths between all pairs of nodes with the restriction that they all must pass through the node A.

How to submit?

Label the source file 'hw5.py' and the report report5.pdf. Submit the report on canvas and the source at

https://teach.engr.oregonstate.edu/teach.php