CS 461 – ARTIFICIAL INTELLIGENCE

HOMEWORK #4 (5%)

Assigned: Thu Mar 29, 2018

Due: Tue Apr 17, 2018, **2:00 pm** NOTE THE UNUSUAL DUE DATE!!!

You can do this homework in groups. Do not forget to indicate the students submitting the homework (at most 5 names). Submit your homework to our TA. (He'll send you instructions about this soon.) Feel free to use any programming language as long as any of you can give a demo using your notebook when requested. The usual late submission policy applies.

The Fifteen Puzzle is a game in which there are 15 1x1 tiles arranged on a 4x4 square so that there is one 1x1 uncovered (empty) area on the square. The tiles are labeled 1 through 15, and initially they are arranged in some random order. The idea is to reach the goal state (which has the southeast corner square empty, as shown below) from a given initial state by moving tiles one at a time.

3	2	1	4
5		11	8
9	7	10	12
13	14	6	15

S

→

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

G

Implement the A* algorithm to solve a given instance of the Fifteen Puzzle optimally (making the smallest number of tile moves). Here's what you must do:

- Write a 'puzzle generator' first. Starting from the goal state of Fifteen Puzzle, the generator returns a reasonably garbled initial state (S) by randomly shuffling the puzzle. Notice that the generator thus guarantees that S will be solvable.

- Run your generator as many times as necessary to obtain 25 distinct initial states.
- Implement the A* algorithm (cf. Winston, ch. 5). You cannot verbatim use an existing implementation found elsewhere. Your implementation should be made either from scratch or with some modifications of an existing code, and solely by your group members. If you are modifying someone else's code, make sure to describe, in detail, your own contributions. Our TA reserves the right to deduct as many points as warranted if your code is not much different from an existing program.
- You must do loop checking and may want to use dynamic programming.
- Run your A* program with those 25 initial states. Your program should employ two heuristics:

  h1: number of tiles-out-of-place heuristic
  h2: sum of Manhattan distances heuristic

  (These admissible heuristics are explained in any reasonable place on the Net.)

A submission consists of

- code listing
- one sample solution, for a particular S, using h1 (*this must be a visual sequence of puzzle snapshots, starting with S and reaching G*)
- repeat the previous step using the same S but h2 instead of h1
- snapshots of 25 initial states of the puzzle (Next to each snapshot write the number of moves to solve it, as calculated by your program. Both h1 and h2 should give the same number. Beware!)
- a plot

The plot shows time behavior of A* vis-a-vis the two heuristics. The x-axis goes from 1 to 25. In order to measure the CPU time taken by a program, invoke system timer before you call it and stop the timer when it terminates. (This may not be a very precise measure but we don't care.) Use different colors or markers (e.g., + and -) to differentiate between h1 and h2. Do not forget to state the time unit you're using.

CAVEAT: ***The goal state of the Fifteen Puzzle can be reached from any solvable instance within 80 moves or less. It would be wise for your puzzle generator not to shuffle too much on pain of getting hard-to-solve (resource demanding) instances.***