# CS3641 Homework 02 Summer 2024

# 8-Tile Puzzle Agent

**Total Points: 100**

Be as brief as possible and use your own words when describing concepts.   SHOW ALL WORK for Questions requiring calculations and algorithms. IMPORTANT!!! - See bottom for instructions on submitting code (I need it in text format!).

To implement Depth-First Search (DFS), Uniform-Cost Search (UCS), Best-First Search (BFS), and A\* Algorithm to solve the following 8-puzzle problem (i.e. find the goal):

8-puzzle Problem:

The 8-puzzle consists of eight numbered, movable tiles set in a 3x3 frame. One tile of the 8-puzzle is always empty thus making it possible to move an adjacent numbered tile into the empty tile position. Start with a random state (Do not make it a fixed starting state). The goal state is the following pattern [1,2,3,8, <empty>, 4,7,6,5] empty tile in center of 3x3 square:

1 2 3

8 4

7 6 5

The agent goal is to move from the initial configuration to reach the goal configuration after an acceptable amount of moves (i.e. not brute force). You must write your own code for the agent and its algorithms. You can represent blank as the value 0.

Your program should be able to address any initial configuration, i.e. not a hardcoded or rules based algorithm/agent and provide a table of statistics below in your PDF file.

You must write your own code for this assignment. Standard libraries are acceptable.

Submit your responses in this document below (except for the source code, see bottom of assignment regarding code).

Q-1: Provide your information:

**// Student name:**

**// Student ID:**

**// The assignment #:**

**//**

**// Signature of source code author: (you can replace text here typing your name)**

**5 points**

Q-2: Using the output data on multiple runs; populate the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm** | **Best, Average, Worst number of nodes visited** (you need repeat each algorithm several times with  different initial configuration) | **Give the best run time, the worst run time, and average run times from data on five or more runs.** | **Comments on Algorithm** |
| **DFS** | Best:  Worst:  Average: | Best:  Worst:  Average: |  |
| **UCS** (using depth as cost) | Best:  Worst:  Average: | Best:  Worst:  Average: |  |
| **BFS** (using  Manhattan distance as heuristic) | Best:  Worst:  Average: | Best:  Worst:  Average: |  |
| **A\*** (using the tiles out-of place as the heuristic) | Best:  Worst:  Average: | Best:  Worst:  Average: |  |

**30 points**

Q-3: Provide the build instructions and source code in a **separate** ***text*** file in D2L (i.e. one of .py, .cpp, .c, cs, .java, or other language file in text or even just .txt with your code in it; see bottom of assignment)

**50 points**

Q-4: Provide a screen shot of your built source code in action on an arbitrary configuration including any execution command (e.g. ./puzzle, or ./hw02 etc.) and its input and output.

**15 points**

**For example here is some example output that would help collect statistics:**

*1) DFS*

*2) UFS*

*3) BFS*

*4) A\**

*5) Exit*

*Choose Algorithm: 2*

*Enter Number of Trials: 5*

*Random Starting Puzzle:*

*[[2,7,3]*

*[6,4,0]*

*[1,5,8]]*

*Goal:*

*[[1,2,3]*

*[8,0,4]*

*[7,6,5]]*

*Algorithm: 2*

*Iteration 1: Left*

*[[2,7,3]*

*[6,4,0]*

*[1,5,8]]*

*Iteration 2: Down*

*[[2,7,3]*

*[6,0,4]*

*[1,5,8]]*

*...*

*Iteration 25: End*

*[[1,2,3]*

*[8,0,4]*

*[7,6,5]]*

*Runtime: 821.23 ms*

*Nodes Visited: 43933*

*…*

*<Repeats 5 times…>*

*…*

*Data for Nodes Visited: 38652, 12456, 49111, 43933, 75184*

*Minimum: 17915*

*Average Visited: 43867.2*

*Maximum: 75184*

*Data for Runtime: 732.15ms, 112.19ms, 301.98ms, 821.23ms, 613.44ms*

*Minimum: etc…*

*Average: etc…*

*Maximum: etc…*

**Submission Guidelines:**

* No handwritten submission is accepted, always submit answers as text within this or similar document file with any support images embedded in the file.
* **EXCEPTION**: If asked for source code implementation you can submit those individually and as separate files in ASCII format in their original file format .cpp, .java, .py, .cs etc. or even as a .txt file will be acceptable. Do not insert code into the submission document file. It ruins spacing which makes .python and some languages (perl, awk etc.) difficult to test build.
* Do not submit ZIP files… ever… for anything in D2L. The system is extremely unhelpful with regards to those filetypes and grading.
* You may include your freehand drawing/image and handwritten scans in the submission. However, the writing and images must be clearly legible. Though, it is best to present non-handwritten submissions, generally, as is done in the professional setting.
* If asked, show all work/calculations/graphs etc. in the determination of the problem.
* **Please complete your entire work in a single Word Document and Save the file as: yournetid\_CS3502\_Assignment01.docx (e.g. ogarcia5\_CS3502\_Assignment01.docx.) and upload your file in D2L.**
* Please observe the submission due date and time. After the due date there is a 50% penalty for the next 24 hours. Any submission after 24 hours of the due date will be graded at 0%.
* If you include a reference or an image taken from other sources, please cite them appropriately. APA is preferred but cite them so they can be found. **NOTE: verbatim copying or even paraphrasing is plagiarism so if the source used constitutes your answer rather than simply *supporting* the answer, it will be considered invalid. This is especially true of source code implementation answers.**
* If you resubmit, please make sure to attach the file again. Your latest submission before the due date will be the one graded.