**Artificial Intelligence (Lab)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ex. No** | **Experiment** | **Dataset** | **R shiny App** | **Due Date** |
| 1 | Using BF, DFS and IDS find a path from a source to goal node and do (plot) the time complexity analysis. | <https://snap.stanford.edu/data/>  <https://archive.ics.uci.edu/ml/datasets.php>  <http://konect.cc/networks/>  Note: Don’t take very large network | Yes | 27-02-2020 |
| 2 | Using BFS, DFS and IDS find the spanning tree, presence of cycle in networks and do (plot) the time complexity analysis | -do- | Yes | 6-03-2020 |
| 3 | Consider the problem of finding a path in the grid shown in the following from the position **s** to the position **g**. A piece can move on the grid horizontally or vertically, one square at a time. No step may be made into a forbidden shaded area.   1. On the grid shown in the figure number the nodes expanded (in order) for a DFS, BFS and IDS from **S** to**g**, given that the order of the operators is up, left, right, and down. Assume there is cycle pruning. What is the first path found? 2. For a greedy best-first search from **s** to **g**. Manhattan distance should be used as the evaluation function. The Manhattan distance between two points is the distance in the x-direction plus the distance in the y-direction. It corresponds to the distance traveled along city streets arranged in a grid. Assume multiple-path pruning. What is the first path found? 3. For a heuristic depth-first search from s to g, given Manhattan distance as the evaluation function. Assume cycle pruning. What is the path found? 4. Number the nodes in order for an A\* search, with multiple-path pruning, for the same grid. What is the path found? | https://artint.info/2e/html/x373.png | Yes |  |
| 4 | **Comparing Algorithms** : Algorithms can be compared on   * the time taken, * the space used, and | yes |  |  |
| 5 | Solve the 8-puzzle problem, using DFS, BFS and IDS and Comparing Algorithms w.r.t time and space | yes |  |  |
| 6 | The A\* search can be used to solve the 8-puzzle problem. There are two candidate heuristic functions: (1) h1 = the number of misplaced tiles; (2) h2 = the sum of the distances of the tiles from their goal positions.  You are to implement the A\* using both heuristics and compare their efficiency in terms of depth of the solution and search costs.  The following figure provides some data points that you can refer to. To test your program and analyze the efficiency, you should generate random problems (>100 cases) with different solution lengths.  To help the instructor evaluate your program, the input of your program should be either (1) a randomly generated 8-puzzle problem by your program; or (2) a specific 8-puzzle configuration entered through the standard input, which contains the configuration for only one puzzle, in the following format:  1 2 4 0 5 6 8 3 7  The goal state is:  0 1 2 3 4 5 6 7 8  Where 0 represents the empty tile | yes |  |  |

