AI - Assignment 4

Amir H. Pakdaman: amirhossein.pakdaman@smail.inf.h-brs.de

Due Date: 13.12.2020

0.1 Programming Assignment: 8-Puzzle game

During the lecture you have discussed two heuristics for the 8-puzzle: **Manhattan distance** and **misplaced tiles**. Your tasks for this week are:

- Implement a Greedy and A* agent for the 8-puzzle. The agents should be able to switch between both heuristics. Make sure to produce proper output to "visualize" the working of your program.
- Compare the performance of the solvers and the two heuristics. Provide data in your report to support your arguments (number of visited nodes, path cost, execution time, etc). Which works better?
  - Use the following initial configurations:
    $\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 8 & 6 & 7 \end{bmatrix}$, $\begin{bmatrix} 8 & 7 & 6 \\ 5 & 1 & 4 \\ 2 & 0 & 3 \end{bmatrix}$, $\begin{bmatrix} 1 & 5 & 7 \\ 3 & 6 & 2 \\ 0 & 4 & 8 \end{bmatrix}$
  - Please note a goal configuration would mean:
    $\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}$

2. Answer the following questions regarding A* search:

- When is A* complete?
- When does A* end the search process?

0.2 Implementation Details

- The implementation of the 8-puzzle game board has already been done in the *Puzzle* class. You may use the methods of the class for developing other parts of your code.
- *get_possible_moves* is used to find new configurations of the boards by checking for possible up, down, left, right moves and adding them to the heap.
- Functions *move_up, move_down, move_left, move_right* are used to check if a swap is possible between the 0 tile and any tile up, down, left, right respectively. If so the swapped configuration is returned.
- *misplaced_tile_heuristic* and *get_no_of_misplaced_tiles* should be used to implement the misplaced tiles heuristic. (optional to split the implementation between these two functions or use a single function)
- *manhattan_distance_heuristic* and *get_manhattan_distance* should be used to implement the Manhattan distance heuristic. (optional to split the implementation between these two functions or use a single function)
- The search algorithms should be implemented in *Greedy_search* and *Astar_search*.
- Please note that **0 represents the empty tile**. Swapping is allowed only with the empty tile.

- For more details regarding changing the heuristics while running your code, please refer to the travis.yml for command line options
- Please add comments explaining your code as and when required. Also fill in the docstrings for the implemented functions.
- You may add additional functions if required and modify the parameters of the existing functions.
- Any extra functions required as per your implementation should be included in helper.py. Please add the necessary docstrings and keep your code as clean as possible.

0.3 Deliverables

- Add 12 text files to the *results* folder, for each initial configuration, each heuristic, and for both A* and Greedy agents. Files should contain the traceback from the initial configuration to the goal configuration. Plots should be in the shape of a puzzle. (3x3, not 1x9)
- Write the execution times in a table. Add it to the results folder.
- Write your conclusion on comparing the performance of the solvers. Add it to the results folder.
- Note that between each step in the traceback, only zero should be displaced, and zero cannot jump, it can move only (up, down, left, right) neighbor cells.
- Codes should run with no errors.
- Codes should follow the theory of algorithms.