Beihang University
School of General Engineering

Beijing, May 17th, 2021
Wei Lin and Yida Ding

Data Management and Artificial Intelligence Homework 11

Task 1 *N-Puzzle*

Implement a class *State* for N-Puzzle, which models the game of playing a puzzle with N tiles. For instance, 8-Puzzle is played on a board with 3*3 slots and the next larger puzzle is 15-Puzzle which is played on a board with 4*4 slots. The number N should be passed to the constructor of *State* along with the initial configuration.

1. Implement functions for goal state judgement, legal action identification, state transition and etc. Keep in mind that your code should work for any valid N!

2. Implement the heuristic function, i.e. Manhatten distance, to evaluate the h value for a specific state.

3. Implement a function which computes an initial configuration of the N-Puzzle at most k steps away from the goal state. **Hint:** Just simulate moving the empty tile around for k steps from the goal state, avoiding previously seen states.

4. Implement three uninformed searching methods: BFS, DFS and UCS (Uniform-Cost Search) to solve the initial configuration generated in subtask 3. **Hint:** You may consider to use *heapq* package to implement priority queue.

5. Implement two informed searching methods: GBFS (Greedy best first search), and A^* to solve the initial configuration generated in subtask 3.

6. Visualize the number of expanded nodes for different values of $N \in \{8, 15, 24, 35\}$, $k \in \{3, 6, 9, 12, 15\}$ and searching method $m \in \{BFS, UCS, GBFS, A^*\}$ using matplotlib. A possible outcome is visualized in Figure 6.

7. Visualize the runtime for different values of $N \in \{8, 15, 24, 35\}$, $k \in \{3, 6, 9, 12, 15\}$ and searching method $m \in \{BFS, UCS, GBFS, A^*\}$ using matplotlib. A possible outcome is visualized in Figure 7.

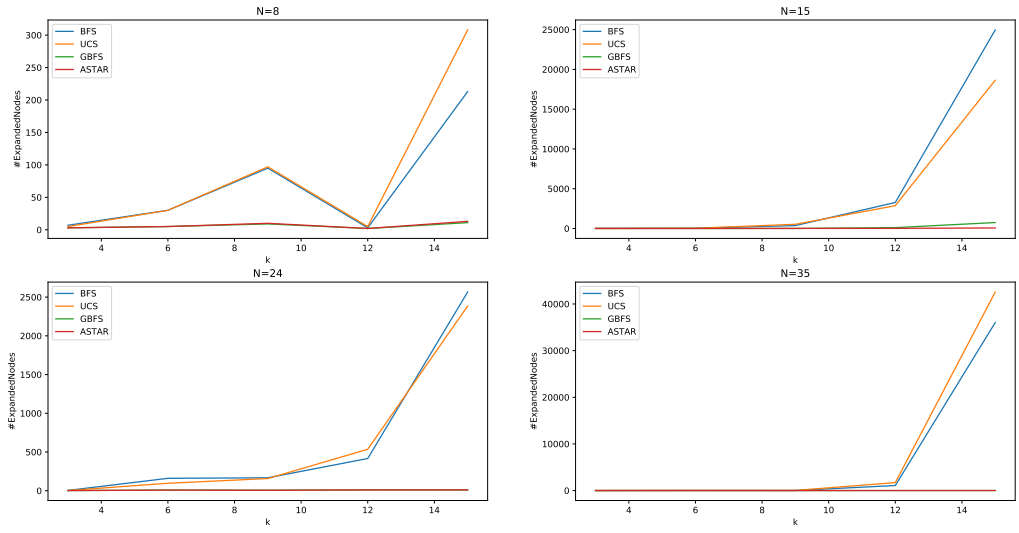

Figure 6: Task1-1

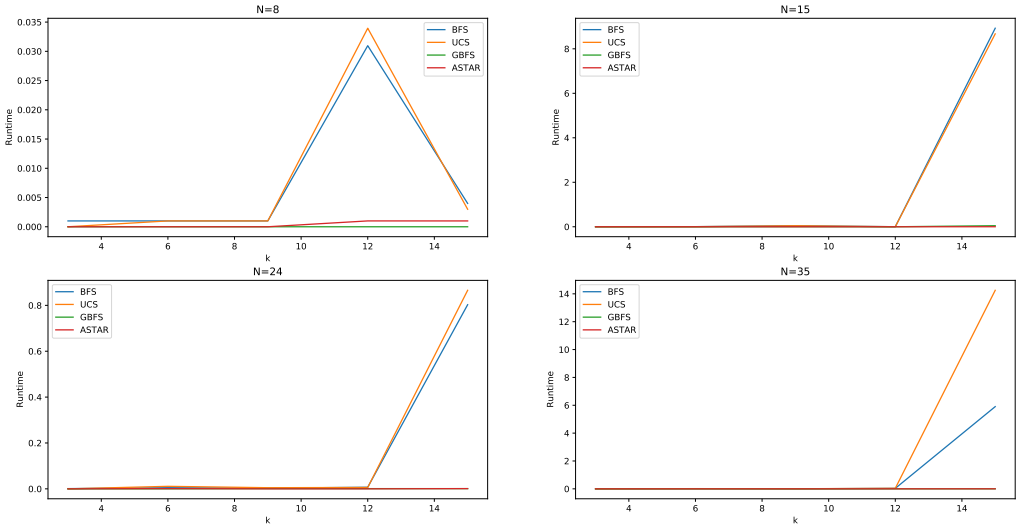

Figure 7: Task1-2