AI Homework 2

Steven DiCarlo snd7nb
Yaser Qazi yq4du

1.4 State has two elements: the priority queue (pq) and the list of seen_puzzles.

The priority queue contains sequences of puzzles, where each puzzle follows from the previous one. The priority is determined by how many steps this sequence is plus the number of tiles still out of place in the final puzzle of the sequence.

In each transition between states, the top priority sequence is removed from pq. Each possible puzzle that can be generated in one move from the last puzzle in the sequence is then generated (call them successors). For each successor, if it is the goal state, then that chain is the solution. Otherwise, for each successor not in seen_states, it is added to seen_states and the sequence ending in it is added to pq.

This is adequate because every possible puzzle that can be generated from the original will eventually be checked. Once pq is empty, every possible puzzle will be in seen_states, none of which were the solution, so there could not have been any solution

1.5 Proving correctness: If it returns something, was it right?

Case 1: returns solution

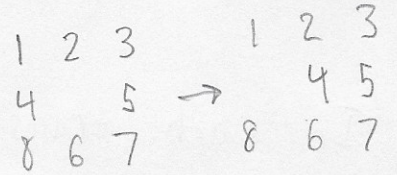
·If it returns a solution, that means it could find a chain of moves that led to the goal state, which means it was correct

Case 2: returns unsolvable:

·At the beginning, we check if the puzzle is solvable by counting the number of inversions. Inversions are when if you list the puzzle in order (left to right top to bottom, like reading a book), every time a bigger number precedes a smaller one is an inversion. We claim that if you start with a puzzle with even inversions (even puzzle), every move will result in another even puzzle.

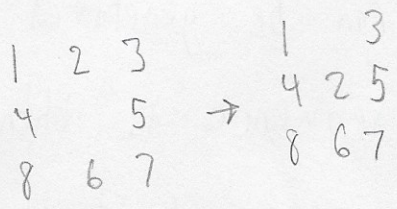
Case A: row move:

If the move shifts a tile along its row, then the order of the tiles is the same, so there is the same number of inversions, so it is still even.



Case B: column move:

If the move shifts a tile along its column, then that tile moves relative to two tiles in the order (in this example 2 goes from before 3 and 4 to after them). This means that the number of inversions can only change by -2, 0, or 2 with each column move, so if it started even, it will stay even.



Since the goal state has an odd number of inversions (7), that means it can't be obtained from any even puzzle, so if our algorithm returns unsolvable it is correct.

1.6 Proving completeness: If there is a solution, will we find it?

In order for there to be a solution, it must be generated from a sequence of moves from the original puzzle. Our algorithm generates all possible puzzles from the original, and eventually checks all of them to see if it is a goal state. Therefore if there is a solution, our algorithm will eventually be able to find it.