Expt 9 David Daniels - 20232000038

Div A Batch C

This experiment on solving 15 puzzle problem has been solved in c++ by using Branch and bound algorithm. Branch and bound works by checking all possible cases for movements (up / down / left / right) and check if it's distance (manhattan distance) to the goal array increases or decreases. If it decreases, it continues with that path otherwise it drops it and works on other possible paths.

This algorithm uses BFS using a priority queue to traverse across the states calculated.

When the distance to goal array is 0, algorithm terminates.

There is a visited state array to prevent the algo from revisiting states reached by another path.

The time complexity of this algorithm is exponential. In the worst case, its time complexity is 16! (16! is checking every single state of the board).

The average case time complexity depends heavily on how good is the distance to goal function defined.

The priority queue is used to prefer taking states with less distance first. It's time complexity is in a log n time and can be ignored.

The branch and bound algorithm gives a systematic approach to solve the 15 puzzle problem. But it's worst case time complexity makes it unsuitable for bigger problems. The effectiveness depends on the quality of distance function