Lab Report
DA221 - Introduction to Artificial Intelligence

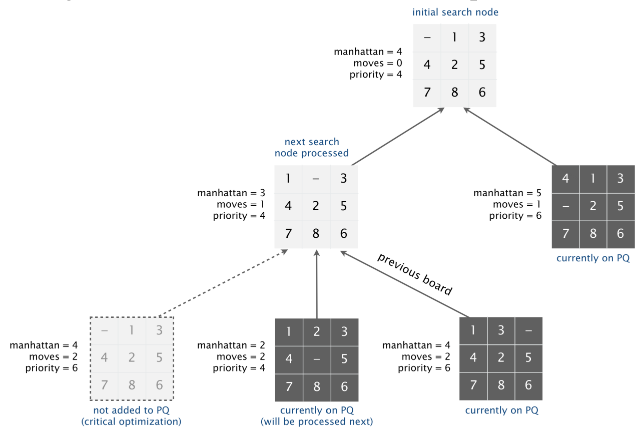
Rishita Agarwal 220150016

1. Introduction

The 8-puzzle problem is an example of a sliding block puzzle. The goal of the puzzle is to move the tiles in a 3x3 grid to achieve a particular goal state from a given initial state, using the least number of moves. This document presents a detailed evaluation of three prominent heuristic techniques - the Hamming distance, the Manhattan distance, and the Linear Conflict heuristic - each offering a unique approach to estimate the 'closeness' of a given puzzle state to the goal state.

2. The 8-puzzle Problem

The puzzle starts with a random initial state, with 0 representing the location of the missing tile. Our goal is to make moves such that the initial state can be converted in a desired goal state. I made the computer generate 10 random samples of valid initial and goal states. The validity of the initial state in a **8-puzzle problem** is checked by a well known algorithm of counting inversions. The following is an image of how a move can be made on a 8 puzzle-



Solving an 8 puzzle is like a trial and error game.Here we try different moves which a particular state can have and choose the one which gives us the most optimal path. **Optimal path** is the path with the least effort or cost to reach the goal state.

2.1. Solvability of the initial state

I counted the number of inversions in the vector of initial state. If the **number of inversions are even** then, the puzzle is solvable, else it is not solvable. This comes from the fact that the goal state has zero inversions (as all the tiles are in order), and since the game allows you to swap the blank with adjacent tiles (which either preserves the inversion count or changes it by an even number), you can reach the goal state through a series of moves that preserve the **even parity**.

3. Heuristic Function

A* Algorithm: **f(n)=g(n)+h(n)**, where h(n) is the heuristic function, g(n) is the cost to reach the current state from the initial state and f(n) is the total cost.
Heuristics simplify the decision-making process by reducing the complexity of the problem, allowing for a quicker decision or estimation.Heuristics in problem-solving, like those used in algorithms, are informed guesses that aim to find a good enough solution or to get closer to an optimal solution. For the 8-puzzle, I implemented 3 different heuristics - **Hamming Distance, Manhattan Distance and Linear Conflict**. I also printed their time taken, the number of nodes they removed from the frontier and the minimum number of steps taken to reach the **goal state** from the **initial state**.

3.1. Hamming Distance

Hamming distance is one of the most basic heuristics. It can be useful for giving a quick, rough estimate of puzzle complexity, but it can be overly simplistic. According to the results printed in the code, we observed that the number of nodes removed from the frontier are very large as compared to manhattan and linear conflict heuristic. Hamming distance only considers whether a **tile is in the correct position or not**, without taking into account how far a misplaced tile is from its correct position, it often underestimates the true cost to reach the goal. This can lead to more nodes being added to the frontier as the search algorithm explores various paths that seem equally promising according to the Hamming distance but may actually be far from optimal. This also increase the search time of this heuristic.

3.2. Manhattan Distance

Manhattan distance is a **more informed heuristic** because it considers the actual number of moves each tile needs to get to its correct position. This generally results in a search algorithm that expands fewer nodes since it can more accurately prioritize which nodes to explore next. As a result, fewer nodes are removed from the frontier compared to Hamming distance, leading to a **more efficient search**. Thus, this complements with the results obtained from our code, that this heuristic works better than hamming distance, in terms of time taken and number of nodes removed from the frontier both.

3.3. Linear Conflict

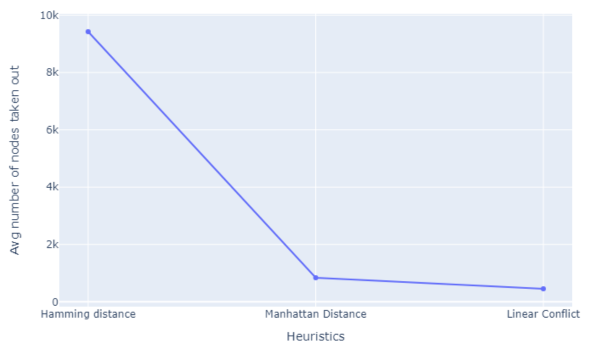
Linear conflict **adds even more information** by considering not only the Manhattan distance but also the arrangement of tiles that are in the correct row or column but out of sequence. This adjustment to the heuristic can often discern when the Manhattan distance might be artificially low due to these conflicts. In such cases, the Linear Conflict heuristic provides a **higher cost estimate that more closely reflects the true cost**, avoiding the expansion of nodes that appear promising according to Manhattan distance alone but are actually not due to these conflicts. Consequently, even fewer nodes are removed from the frontier, making the search process potentially more efficient than with Manhattan distance alone.

4. Evaluation

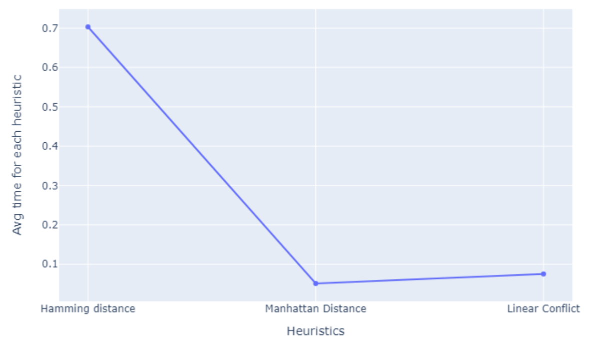
Heuristic	Avg time Taken (in s)	Avg No. of Nodes Taken Out	Accuracy
Hamming	High	High	Low
Manhattan	Low	Medium	Medium
Linear Conflict	Medium	Low	High

Experimental Results on the 10 random instances

The graph shown below is to compare the average number of nodes taken out over 10 different sample examples for the 3 heuristics. We can clearly see that the average nodes taken out using Hamming distance are much more as compared to those taken out by Manhattan and Linear Conflict. This is because hamming distance takes only the number of misplaced tiles into account, without taking into account the actual distance from the goal state.Linear conflict is having least average nodes taken out because it takes more factors into consideration for a better or a more closer state selection to reach the goal state.



The graph shown below is to compare the time taken by Hamming distance is maximum again due to its overly simplistic approach whereas linear conflict here takes more time than manhattan distance due to the use of manhattan and an extra check for linear conflicts over the different nodes. These results match the algorithms that the heuristics follow.



Heuristic	Avg time Taken (in s)	Avg Number of Nodes Taken Out
Hamming	0.703322	9425
Manhattan	0.0510848	841
Linear Conflict	0.075122	454

Experimental Results on the 10 random instances

5. References

- Principles of Artificial Intelligence
- Princeton page on 8-Puzzle
- 8 puzzle problem on GFG