Dean&Francis
ISSN 2959-6157

Solving the Sliding Puzzle problem using the A* algorithm and comparing the effectiveness of different heuristic functions

Zike Qin[1,*],

Mingfei Zhang[2]

[1]The Experimental High School Attached to Beijing NormalUniversity, Beijing, 100082, China, snow20170601outlook.com
[2]Queen Anne's School, Reading, RG46DX, United Kingdom, mingfeiz36@gmail.com

Abstract:

This paper aims to solve the Sliding Puzzle problem using the A* algorithm and compare the effectiveness of different heuristic evaluation functions. Multiple heuristic functions including Manhattan Distance and Linear Conflict will be investigated. This helps to determine the best heuristic function of the A* algorithm by considering efficiency and accuracy.

Keywords:- Sliding Puzzle, A* Algorithm, Heuristic Search, Manhattan Distance, Linear Conflict

1. Introduction

The Sliding Puzzle problem is s classic combinatorial optimization problem used to study the search capabilities and optimization performance of algorithms. The Fifteen Puzzle, also called Gem Puzzle, was a sliding puzzle invented by Noyes Palmer Chapman in 1874 [1]. It consists of 15 square tiles numbered in numerical order from 1 to 15 in a $4*4$ grid with one empty square left, which forms a subgroup inside S_{15} [2]. It is stated that each tile is only allowed to slide to the adjacent space. The puzzle aims to obtain goal board positions by rearranging the tiles using legal actions. To solve the Fifteen Puzzle, even permutations are involved. Every permutation is a product of transpositions. For instance, when we cycle decomposition, (123) is even as (123) = (12)(23) which has two transpositions, while (1234) is odd as (1234) = (12)(23)(34) which has three transpositions [3]. According to the rules of the Fifteen Puzzle, we know that every move is a product of transposition involving the empty slot. If we want to swap the positions of numbered tiles and keep the space in the same place, ultimately, the number of up and down or left and right transpositions must be the same. Thus, we can deduce that if a puzzle board is solvable (able to create possible configuration), it needs to be corresponding with an even permutation of S_{15}.

In the late 1870s, an American mathematician, Sam Loyd, launched a challenge with the prize aim of swapping the 14^{th} and 15^{th} tiles to end up with the standard starting position [1]. However, (14,15) is an odd permutation, which means there is an odd number of transpositions. This contradicts the fact that solving the Fifteen puzzle needs an even permutation which then shows that the problem is impossible to solve. Based on that, we can infer that any two numbers in the Fifteen Puzzle are not allowed to swap directly as there is an odd permutation.

To find the optimum solution for the Sliding Puzzle, we are going to review two traditional algorithms: the A* Search Algorithm and Greedy Best First Search.

ISSN 2959-6157

A* Search Algorithm is a heuristic search widely used in various pathfinding and problem-solving tasks due to its efficiency and flexibility. It uses the least cost to find the shortest route of the solution. It can be written as f(n) = g(n) + h(n). F(n) is defined as the total cost, g(n) is defined as the cost between the original configuration to the current positions, and h(n) shows the costs from the current to the target configuration [4]. In comparison, Greedy Best First Search is much faster than the A* Search Algorithm as it excludes the g(n). While Greedy Best First Search does not necessarily use the shortest path [4]. Both of the algorithms are efficient in solving the Sliding Puzzle. In this paper, we will only be focused on the A* Search with different heuristic functions.

2. Related Work

Heuristic search algorithms have garnered extensive attention for their efficiency in solving the Sliding Puzzle problem, such as the 15-puzzle. Common heuristic functions include Manhattan Distance and Linear Conflict. Manhattan Distance evaluates the cost of a state by calculating the distance between the current state and the target state. This heuristic never overestimates the cost to reach the goal as it is consistent [5]. However, it does not account for the case that tiles need to be moved more than one time under conflict conditions. While Linear Conflict supplements and optimizes Manhattan Distance. It can identify the inversions in the same row or column. It takes into account the need to move conflicting tiles, which provides a more accurate number of moves than the Manhattan puzzle, which includes additional computation and complexity [6].

3. Methodology

3.1 A* Algorithm Overview

The A* algorithm is a heuristic-based search algorithm that evaluates each node (n) using a heuristic function (h(n)) combined with the actual cost (g(n)) from the start node to form the total cost (f(n) = g(n) + h(n)). The algorithm expands the node with the smallest (f(n)) from the open list until the goal state is reached [4].

3.2 Problem Modeling

The Sliding Puzzle problem can be modelled as a state-space search problem. Each state represents an arrangement of the tiles on the board, and operations include swapping the blank tile with its adjacent tiles. The goal state is when all the numbers are in positive sequence, and the blank tile is in the last position [3].

3.3 Heuristic Evaluation Functions

3.3.1 Manhattan Distance

Manhattan Distance is a common method in heuristic search, calculating the sum of the horizontal and vertical distances of each tile from its target position.

$$h(n) = \sum_{i=1}^{n} \left| x_i - x_i^* \right| + \left| y_i - y_i^* \right|$$

(x_i, y_i): Represents the current position of piece *i*.

(x_i^*, y_i^*): Represents the target position of piece *i*.

3.3.2 Linear Conflict

Linear conflict considers the inversion pairs present in the same row or column based on Manhattan distance. If there are pieces in the same row or column with incorrect relative positions and mutually obstructing each other, it is considered a pair of linear conflicts [5]. Each pair of linear conflicts adds an extra cost of two units:

$$h'(n) = h(n) + 2 \times linearconflicts$$

4. Experimental Results

4.1 Experimental Setup

The scale of the sliding puzzle problem used in the experiment is 4x4. We generated several random initial states and solved them using two heuristic functions: Manhattan distance and linear conflict. The experiment evaluated the number of steps and solving time for each heuristic function.

4.2 Results and Analysis

We conducted tests on 16 solvable random 4x4 puzzles. The raw data is presented in Table 1. After processing the data, we obtained comparisons of the computation times required by two heuristic functions under different step counts, as shown in Figure 1. Additionally, Figure 2 illustrates the relationship between the number of inversions and the computation time.

Table 1 16 solvable random 4x4 puzzles

Test	Step	Time (Manhatten Distance)(s)	Time (Linear Conflict)(s)	Number of Inversions
1	38	22.082	12.819	18
2	20	0.095	0.051	20
3	44	160.451	14.144	36
4	48	2047.714	784.195	58
5	43	175.855	69.524	38
6	44	410.700	191.592	28
7	44	2104.952	893.230	42
8	51	4295.814	2261.206	49
9	51	8106.868	801.277	40
10	47	3029.457	844.796	48
11	44	96.600	48.335	46
12	48	853.387	788.878	52
13	34	3.277	2.013	36
14	47	3621.418	1972.524	43
15	49	201.308	146.591	42
16	40	137.127	23.272	42

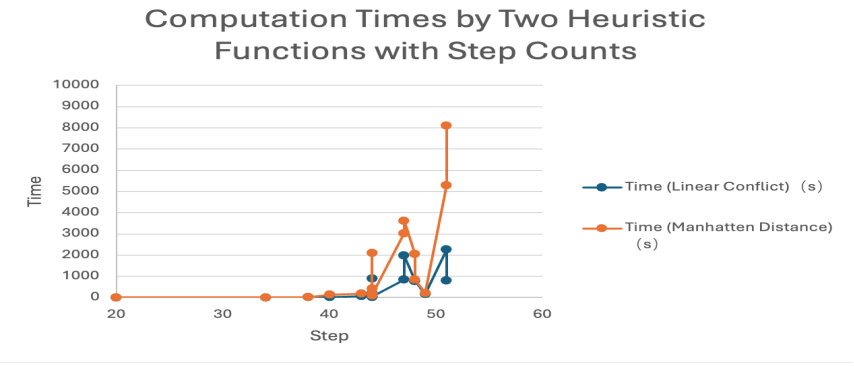

Figure 1 Comparisons of the computation times required by two heuristic functions under different step counts

It can be observed in Figure 1 that the two lines in the above figure have a similar shape. However, the line representing the heuristic function based on linear conflict is generally lower than the line representing the Manhattan distance. Both lines exhibit an overall exponential growth trend.

ISSN 2959-6157

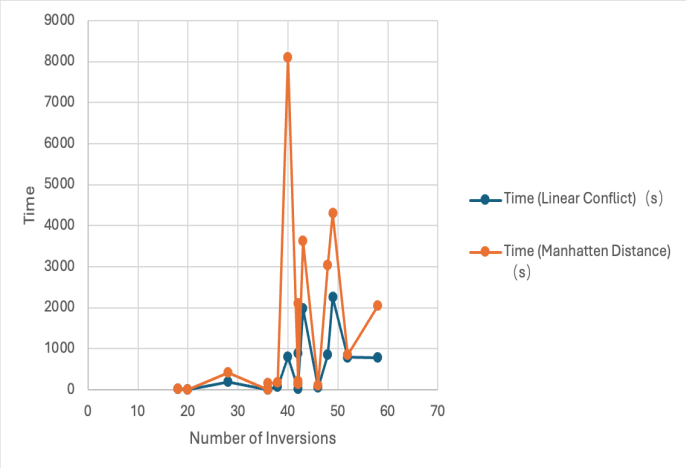

Figure 2 relationship between the number of inversions and the computation time

Although Figure 2 does not visually show a direct correlation between time and the number of inversions, by comparing it with Figure 2, it can be inferred that the peaks in Figure 1, which deviate from the exponential growth trend, correspond to higher numbers of inversions in Figure 2.

The experimental results show that the linear conflict heuristic function outperforms the Manhattan distance in terms of time. This is because linear conflict further optimizes the Manhattan distance, reducing ineffective path expansions. Furthermore, during our testing, we observed that when the initial state had a higher number of inversions, the computation time and the number of steps required were also greater. However, possibly due to the relatively small size of the 4x4 grid, we did not observe any cases where the two heuristic functions produced different optimal step counts for the same random configuration. This divergence may become apparent as the grid size, denoted by n, increases.

5. Discussion

The choice of heuristic function has a significant impact on the performance of the A* algorithm. Manhattan distance is commonly used due to its simplicity and efficiency, but for more complex puzzle problems, its estimates may not be accurate enough. In contrast, linear conflict provides a more accurate estimate by considering additional constraints. Therefore, the choice between the heuristic functions depends on the requirements of the problem being solved. If a more accurate solution cost is desired than computational efficiency, the linear conflict is more suitable to use. Future research can explore other more advanced heuristic functions or hybrid methods combining multiple heuristics to further improve the efficiency of the algorithm.

Additionally, in my own attempts to solve the 4x4 sliding puzzle, I discovered a pattern for swapping diagonal numbers. For instance, to swap the number in the (1,3) position with the number in the (2,2) position, first, move the blank space to (2,1), then move the number from (1,1) to (2,1), the number from (1,2) to (1,1), and then cycle the three numbers within the 2x2 sub-grid until the number originally at (2,2) is in the (1,3) position. Finally, move the numbers at (1,1) and (2,1) back to their original positions. Using this method, most of the puzzle can be systematically restored, leaving only a small portion to be adjusted. I believe this is an interesting method that can potentially transform the calculation of effective paths into a series of repetitive combinations, although it may be challenging to find the optimal solution.

6. Conclusion

This paper demonstrates the solution of the sliding puzzle problem using the A* algorithm and compares the practical effects of Manhattan distance and linear conflict heuristic functions. The experimental results show that the linear conflict heuristic function performs better in terms of path length and computation time. This study provides an effective algorithm reference for solving the sliding puzzle problem and offers a possible direction for future research.

References

[1] Archer, A. F. (1999). A Modern Treatment of the 15 Puzzle. The American Mathematical Monthly, 106(9), 793–799. https://doi.org/10.1080/00029890.1999.12005124

[2] Spitznagel, E. L. (1967). A New Look at the Fifteen Puzzle. Mathematics Magazine, 40(4), 171–174. https://doi.org/10.1080/0025570X.1967.11975789

[3] Chapple, A., Croeze, A., Lazo, M., Merrill, H. An Analysis

of the 15-Puzzle. *Louisiana State University*. https://www.math.lsu.edu/system/files/RP1%20paper.pdf
[4] Setyobudhi, C. T. (2022). Comparison of A* Algorithm and Greedy Best Search in Searching Fifteen Puzzle Solution. International Journal of Innovation Scientific Research and Review, 04(07), 3094-3097. http://www.journalijisr.com/sites/default/files/issues-pdf/IJISRR-941.pdf
[5] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE transactions on Systems Science and Cybernetics, 4(2), 100-107. https://ieeexplore.ieee.org/document/4082128
[6] Korf, R. E. (1985). Depth-first iterative-deepening: An optimal admissible tree search. Artificial Intelligence, 27(1), 97-109. https://www.sciencedirect.com/science/article/abs/pii/0004370285900840