IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 14, No. 3, June 2025, pp. 2423~2432
ISSN: 2252-8938, DOI: 10.11591/ijai.v14.i3.pp2423-2432

The main weaknesses of using Manhattan distance for solving sliding tile puzzles

Mohammed Nayef Al Refai[1], Zeyad Mohammed Jamhawi[2], Ahmed Ali Otoom[3], Adai Al-Momani[4], Hayel Khafajeh[5], Issa Atoum[6]
[1]Department of Software Engineering, Faculty of Information Technology, Zarqa University, Zarqa, Jordan
[2]Department of Computer Science, Faculty of Information Technology, Applied Science Private University, Amman, Jordan
[3]Department of Cybersecurity and Cloud Computing, Faculty of Information Technology, Applied Science Private University, Amman, Jordan
[4]Department of Computer Science, Faculty of Information Technology, Zarqa University, Zarqa, Jordan
[5]Department of Artificial Intelligence, Faculty of Information Technology, Zarqa University, Zarqa, Jordan
[6]Department of Software Engineering, Faculty of Information Technology, Philadelphia University, Amman, Jordan

Article Info

Article history:

Received Oct 25, 2023
Revised Nov 20, 2024
Accepted Jan 27, 2025

Keywords:

Domain node
Heuristic
Manhattan distance
Optimization
Pathfinding
Slide tiles puzzle

ABSTRACT

Heuristics are a big improvement over blind searching in pathfinding. The node's test, run, and finish time are reasonable. Artificial intelligence (AI) uses Manhattan distance (MD), a good and simple heuristic, in various subjects to reduce the number of exploring nodes while requiring fewer calculations. The MD heuristics examined approximately 25 times fewer states than the blind search. Unfortunately, can't reach the goal of pathfinding when the domain size increases, as it becomes similar to brute force or blind search algorithm results. Previous studies have concentrated on MD's weakness, specifically its low bound value for calculation results, and attempted to improve this value in various ways. Unfortunately, to our knowledge, none of the presented research has been able to find the optimal path for all slide tile puzzle sizes. This work discusses the detailed reasons for the low bound value and other related factors that contribute to its weakness. This paper discovered that the distribution of MD values within the domain, not lowbound values, is the critical issue that complicates the search. The MD's summation method for all tiles has an impact on the calculated duplication values. The total number of nodes in the optimal path also affects the search performance.

Corresponding Author:

Mohammed Nayef Al Refai
Department of Software Engineering, Faculty of Information Technology, Zarqa University
Zarqa, Jordan
Email: refai@zu.edu.jo

1. INTRODUCTION

The Manhattan distance (MD), also known as the taxicab distance or L1 distance, is a measure of the distance between two points in a grid-like system where movement is restricted to horizontal and vertical directions. It is named after the grid-like street layout of Manhattan [1], [2]. The MD between two points (x1, y1) and (x2, y2) is defined as: $|x1 - x2| + |y1 - y2|$; where "|" represents the absolute value. The result is the sum of the absolute differences between the x-coordinates and the y-coordinates of the two points [3].

For example, the MD between (1, 2) and (4, 6) is: $|1 - 4| + |2 - 6| = 3 + 4 = 7$. Note that the MD is always greater than or equal to the Euclidean distance (the straight-line distance between two points), and it is commonly used in applications such as image processing, computer vision, and machine learning [4], [5]. MD is a good heuristic used in many types of research fields like data mining [6], [7], machine learning [8], face recognition [9], and pathfinding [10]. It is simple and needs a small amount of calculation. Also, it appears better in many research compared to other heuristics like Euclidean distance [9]. MD heuristic is used to calculate the absolute difference between two values as in (1). When there are many values associated with each other [11], [12], MD will be the sum of them as in (2). An example for MD sum is slide tile puzzles which will calculate each equivalent pair in different locations for the two nodes and then sum the result for all tiles.

$$MDi = |y_i - x_i| \tag{1}$$

$$MD = \sum_{i=1}^{n-1} |y_i - x_i| \tag{2}$$

Unfortunately, MD is not practical alone to find a goal for a large sliding tile puzzle. As the desired quality bound produce will not accept the solution as quickly as possible [13], other heuristics related to the domain used with MD in slide tile puzzles like misplaced tile, and linear conflict (LC) [14]. Previous studies focus on MD weakness related to the low boundary value generated at the calculation for actual solution cost [15]. Some studies focus on improving MD by calculating with other heuristics like miss tile and LC [12], [16]. Other studies focus on inventing new heuristics similar to MDs but more effective like walking distance (WD) [17]. Others change the way totally as in the database pattern which provides the best existing admissible heuristics for this slide tile puzzle [18]. The study employs a novel method for hybridizing the Viola-Jones face detection algorithm to track and identify human faces in video sequences using MD measure equations [19].

Slide tile puzzles, also known as sliding puzzles or sliding tile puzzles, are a type of puzzle game where the player must slide tiles or blocks within a confined space to rearrange them into a specific pattern or image. The puzzles typically consist of a grid of square tiles or blocks, with one empty space that allows for the sliding of adjacent tiles. The objective of the game is to rearrange the tiles so that they form a specific pattern or image, often by moving them around in a specific sequence. The puzzles can range in difficulty, with some requiring only a few moves to solve and others being much more complex. Slide tile puzzles have been popular for many years, with some of the earliest versions dating back to the late 19th century. They have since been adapted to a wide range of formats, including electronic games, mobile apps, and online versions. Some popular variations of the puzzle include the 15-puzzle, the 24-puzzle, and the Rubik's Cube. According to Al-Refai and Jamhawi [20], the memory usage in slide tile puzzles was compared using depth-first frontier searches, blind algorithms, and breadth-first frontier searches as an example of a cyclic graph.

The bidirectional search algorithm A* (BA*) with three heuristics, such as LC, MD, and WD, has been used tried to control solve the fifteen puzzle problem in the research article [21]. Authors the large state space. The algorithm is effectively assisted by the three aforementioned heuristics in reducing the number of generated states and expanding fewer nodes. Yiu *et al.* [22] introduced a novel design and optimization method for multi-weighted-heuristics function (MWH) searching algorithms called evolutionary heuristic A* search (EHA*) to reduce the effort on heuristic function design via genetic algorithm (GA), optimize the performance of A* search and its variants, including but not limited to WA* and multi-heuristic A* (MHA)*, and guarantee the completeness and optimality. The primary goal of [23] was to use the snake game as a comparative tool to analyze the variations in search algorithm optimality between human agents and artificial intelligence (AI). This paper focuses on the domain of the slide tile puzzle and the MD calculated value for each node in detail to find the main weakness reason for its value to approve or reject the law boundary reason.

This study will be experimental and rely on slide tile puzzles size 5×2 and 3×3 to be the case study. MD heuristic will be computed by counting the number of grid units that each tile is displaced from its goal position and then summing all tile values without the blank location [18]. MD efficient computation possible simplified problem as individual tiles can move independently of each other [24]–[26].

2. RESEARCH METHOD

The researchers began this research by reading previous research in order to find out the latest developments in science in this field, as explained in the previous section. The researchers then worked on all the domains of the slide tile puzzle for the case study, aiming to determine the node level, path, and MD value. The goal node was located at level 0, which is considered the reference for all paper calculations and results. After all trials and calculations, researchers write down the results and recommendations. The following sub-sections describe the calculations used to obtain the results.

2.1. Domain collection

This research method takes the initial node for the case study as shown in Figure 1 then generates all their related nodes that can be reached by legal moves to extract all the reachable domains. Nodes were generated by using a breadth-first search with frontier boundary [20]. The collected data is saved in an array of arrays for each level node in an orderly way from goal level to maximum reachable nodes level. Figure 1(a) shows slide tile puzzle goal state for 3×3 and Figure 1(b) shows slide tile puzzle goal state for 2×5.

1	2	3
4	5	6
7	8	

(a)

1	2	3	4	5
6	7	8	9	

(b)

Figure 1. Slide tile puzzle goal state for (a) 3×3 and (b) 2×5

2.2. Manhattan distance calculation

MD was calculated for each node in the two domains for the case study comparison between each node and the goal node. Each tile coordinates (x1, y1) in any node compared to its goal node coordinates (x0, y0). The sum for all tiles considers the final MD heuristic result for each node which is used to guide to reach the goal node. The data were collected to check the duplication of values at different levels and its effect on MD heuristic optimal path guiding. Also, extract the first tile from each node generated to find the range of level found on and at the same time MD value for them. The empty tile does not include at MD calculate value because it increases the table MD result range and at the same time increases the range of levels where the same value appears.

2.3. Nodes in each path

All nodes at each level are considered as initial nodes for the previous level then extract the allowed optimal path nodes from them to the goal node. This check will remove the nodes that fall in level maximum than the initial node or even if its neighbor is in the same level there will not be a direct optimal path between them. The local maxima nodes under the initial node level will be removed as they can't consider the optimal path for the node chosen. After removing all local maxima nodes in the first below level, then their parent will become local maxima unless they fall in the initial node paths. This scenario will be repeated until reaches the goal state which will extract the optimal path nodes from the initial state to the goal state. The work done for all nodes falls at a level greater than level 0 for the two domains. The paths extracted for the top-level domains node test for MD to check if it is value optimal through the different levels.

3. RESULTS AND DISCUSSION

3.1. Domain levels result

The domain for the two case studies was collected in levels for each level and the nodes count in it, as shown in Tables 1 and 2. The level count increases until reaching the maximum count at level 24 for slide tile puzzle 3×3, value 24,047, and level 36 for slide tile puzzle 2×5, value 133,107. The domain is a cycle polygon with 12 edges around each node [20].

Table 1. Slide tile puzzle 3×3 levels count nodes

Level no.	Nodes count	Level no.	Nodes count	Level no.	Nodes count	Level no.	Nodes count
0	1	8	116	16	4,485	24	24,047
1	2	9	152	17	5,638	25	15,578
2	4	10	286	18	9,529	26	14,560
3	8	11	396	19	10,878	27	6,274
4	16	12	748	20	16,993	28	3,910
5	20	13	1,024	21	17,110	29	760
6	39	14	1,893	22	23,952	30	221
7	62	15	2,512	23	20,224	31	2

Table 2. Slide tile puzzle 5×2 levels count nodes

Level no.	Nodes count	Level no.	Nodes count	Level no.	Nodes count	Level no.	Nodes count
	1	14	851	28	54,597	42	60,119
1	2	15	1,232	29	65,966	43	45,840
2	3	16	1,783	30	78,433	44	33,422
3	6	17	2,530	31	91,725	45	23,223
4	11	18	3,567	32	104,896	46	15,140
5	19	19	4,996	33	116,966	47	9,094
6	30	20	6,838	34	126,335	48	5,073
7	44	21	9,279	35	131,998	49	2,605
8	68	22	12,463	36	133,107	50	1,224
9	112	23	16,597	37	128,720	51	528
10	176	24	21,848	38	119,332	52	225
11	271	25	28,227	39	106,335	53	75
12	411	26	35,682	40	91,545	54	20
13	602	27	44,464	41	75,742	55	2

The maximum nodes at level 31 for slide tile puzzle 3×3 as shown in Figure 2 and nodes at level 55 for slide tile puzzle 2×5 as shown in Figure 3. They are not fully reverse tiles at the last level as the full reverse found at law level 44 at 2×5 and level 30 for 3×3 so tiles reverse will not be optimal at the levels as the full reverse Manhattan values will be greater than the global maximum level. The increasing of breadth through the levels until reaches maximum breadth is expected to lead to a huge number of paths for any initial node falling in high level but this expectation will not be correct when extracting optimal nodes paths for every node to goal node.

3.2. Manhattan distance results

MD values calculated appeared at a different level for different nodes which will lead to complicating the guidance of research. Tables 3 and 4 show the distinct MD heuristic calculated value at each level for the case studies. MD sum ranges from 0 to 22 for slide tile puzzle 3×3 and 0 to 31 for slide tile puzzle 2×5. The reason for this result is because the final result for each node is the sum of all tiles at the puzzle of MD for each node generated is calculated from the goal node is calculated with its level for generating. The table explains the weakness of the MD which to not an optimal calculation result that leads the heuristic to lose the correct path direction through research guidance and complicates research. When the MD boundary increases with the same pattern then the same result will appear because it is related to the sum of tiles values. Figure 4 presents the relation between level number (Figure 4(a)) and Manhattan value (Figure 4(b)) for Tables 3 and 4 consequently.

6	4	7
8	5	
3	2	1

8	6	7
2	5	4
3		1

Figure 2. Maximum level nodes for slide tile puzzle 3×3

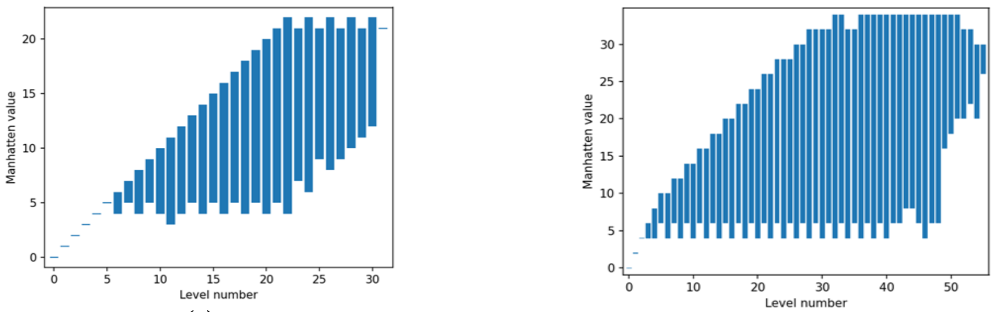
	5	3	2	1
9	4	8	7	6

	9	3	7	1
5	4	8	2	6

Figure 3. Maximum level nodes for slide tile puzzle 2×5

Table 3. MD level heuristic values in slide tile puzzle 3×3

Level no.	MD heuristic values	Level no.	MD heuristic values
0	[0]	16	[4, 6, 8, 10, 12, 14, 16]
1	[1]	17	[5, 7, 9, 11, 13, 15, 17]
2	[2]	18	[4, 6, 8, 10, 12, 14, 16, 18]
3	[3]	19	[5, 7, 9, 11, 13, 15, 17, 19]
4	[4]	20	[4, 6, 8, 10, 12, 14, 16, 18, 20]
5	[5]	21	[5, 7, 9, 11, 13, 15, 17, 19, 21]
6	[4, 6]	22	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22]
7	[5, 7]	23	[7, 9, 11, 13, 15, 17, 19, 21]
8	[4, 6, 8]	24	[6, 8, 10, 12, 14, 16, 18, 20, 22]
9	[5, 7, 9]	25	[9, 11, 13, 15, 17, 19, 21]
10	[4, 6, 8, 10]	26	[8, 10, 12, 14, 16, 18, 20, 22]
11	[3, 5, 7, 9, 11]	27	[9, 11, 13, 15, 17, 19, 21]
12	[4, 6, 8, 10, 12]	28	[10, 12, 14, 16, 18, 20, 22]
13	[5, 7, 9, 11, 13]	29	[11, 13, 15, 17, 19, 21]
14	[4, 6, 8, 10, 12, 14]	30	[12, 14, 16, 18, 20, 22]
15	[5, 7, 9, 11, 13, 15]	31	[21]



(a) (b)

Figure 4. Relation between (a) level number and (b) Manhattan value

Table 4. MD level heuristic values in slide tile puzzle 2×5

Level no.	Heuristic values	Level no.	Heuristic values
0	[0]	28	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
1	[2]	29	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
2	[4]	30	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
3	[4, 6]	31	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
4	[4, 6, 8]	32	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
5	[6, 8, 10]	33	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
6	[4, 6, 8, 10]	34	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
7	[6, 8, 10, 12]	35	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32]
8	[4, 6, 8, 10, 12]	36	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
9	[6, 8, 10, 12, 14]	37	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
10	[4, 6, 8, 10, 12, 14]	38	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
11	[6, 8, 10, 12, 14, 16]	39	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
12	[4, 6, 8, 10, 12, 14, 16]	40	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
13	[6, 8, 10, 12, 14, 16, 18]	41	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
14	[4, 6, 8, 10, 12, 14, 16, 18]	42	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
15	[6, 8, 10, 12, 14, 16, 18, 20]	43	[8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
16	[4, 6, 8, 10, 12, 14, 16, 18, 20]	44	[8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
17	[6, 8, 10, 12, 14, 16, 18, 20, 22]	45	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
18	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22]	46	[4, 6, 8, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
19	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24]	47	[6, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
20	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]	48	[6, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
21	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26]	49	[16, 18, 20, 22, 24, 26, 28, 30, 32, 34]
22	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26]	50	[18, 20, 22, 24, 26, 28, 30, 32, 34]
23	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28]	51	[20, 22, 24, 26, 28, 30, 32, 34]
24	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28]	52	[20, 22, 24, 26, 28, 30, 32]
25	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28]	53	[22, 24, 26, 28, 30, 32]
26	[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30]	54	[20, 22, 24, 26, 28, 30]
27	[6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30]	55	[26, 30]

Tables 5 and 6 show the final MD results and their counts compared to the levels that appear in them. MD became useless as the level increased so the algorithm needed to discover a huge number of nodes to find better MD values, especially at high levels. Also, the value may lead to a path far away from the goal as levels range far. The same MD value is very high as 3×3 reaches 18 levels and in 2×5 reaches 42 levels. In some cases, the MD value reduces but unfortunately, the level range is not reduced as for MD value 5 at slide tile puzzle 2×5.

One of the tests done by giving weight for each tile by multiplying its MD value with its number to increase the final MD boundary. The values of MD increase and its range become between 0 and 132 for slide tile puzzle 3×3 and it's become between 0 and 195 for slide tile puzzle 2×5. Unfortunately, the range of the values level still high even when MD value improved and still the direction of MD value is critical.

Table 5. Final MD levels range and count values in slide tile puzzle 3×3

MD value	Levels range	MD count	MD value	Levels range	MD count	MD value	Levels range	MD count
0	0	1	8	8-26	3,655	16	16-30	22,180
1	1	2	9	9-27	5,084	17	17-29	14,226
2	2	4	10	10-28	10,999	18	18-30	10,825
3	3-11	10	11	11-29	11,862	19	19-29	5,896
4	4-22	115	12	12-30	21,707	20	20-30	2,790
5	5-21	246	13	13-29	20,040	21	21-31	1,186
6	6-24	695	14	14-30	27,625	22	22-30	204
7	7-23	1,134	15	15-29	20,954			

Table 6. Final MD levels range and count values in slide tile puzzle 2×5

MD value	Levels range	MD count	MD value	Levels range	MD count	MD value	Levels range	MD count
0	0	1	11	11-43	26,208	22	22-54	133,375
1	1	2	12	12-46	43,802	23	23-53	101,274
2	2	3	13	13-45	65,918	24	24-54	69,114
3	3-19	8	14	14-46	92,088	25	25-55	43,281
4	4-46	113	15	15-49	120,996	26	26-54	23,598
5	5-47	273	16	16-50	149,091	27	27-53	11,392
6	6-48	681	17	17-49	173,124	28	28-52	4,418
7	7-45	1,434	18	18-50	188,093	29	31-51	1,332
8	8-44	4,194	19	19-51	191,032	30	32-50	304
9	9-45	8,201	20	20-54	182,643	31	37-49	40
10	10-44	15,682	21	21-55	162,685			

3.3. Path nodes results

The paths from the top level for the two nodes appear in the two-case study extracts to find the common path nodes from the top level to the goal. Tables 7 and 8 show there are common path nodes between them at slide tile puzzle 3×3 from level 0 to level 12 then they will move in separated nodes with total nodes in the path for them 1,009. In slide tile puzzle 2×5 the common path node between them starts from level 0 until 16 then there is no common path between them with total nodes in the path for them 1,825. This number is less than 0.6% of the total nodes in slide tile puzzle 3×3 and around 0.1% in slide tile puzzle 2×5. It is true that the domain is polygon but local maxima is the reason for separated paths and small nodes count in paths.

Table 7. The common node in path for top two nodes in slide tile puzzle 3×3

Level	First path nodes	Second path nodes	All path nodes	Common nodes	Level	First path nodes	Second path nodes	All path nodes	Common nodes
0	1	1	1	1	16	29	29	58	0
1	2	2	2	2	17	26	26	52	0
2	4	4	4	4	18	27	27	54	0
3	7	7	8	6	19	25	25	50	0
4	10	10	14	6	20	25	25	50	0
5	12	12	16	8	21	23	23	46	0
6	15	15	26	4	22	20	20	40	0
7	15	15	26	4	23	18	18	36	0
8	18	18	34	2	24	16	16	32	0
9	22	22	42	2	25	12	12	24	0
10	27	27	52	2	26	11	11	22	0
11	26	26	50	2	27	9	9	18	0
12	27	27	52	2	28	7	7	14	0
13	27	27	54	0	29	5	5	10	0
14	29	29	58	0	30	3	3	6	0
15	28	28	56	0	31	1	1	2	0

Table 8. The common node in path for top two nodes in slide tile puzzle 2×5

Level	First path nodes	Second path nodes	All path nodes	Common nodes	Level	First path nodes	Second path nodes	All path nodes	Common nodes
0	1	1	1	1	28	44	8	52	0
1	2	2	2	2	29	44	8	52	0
2	3	3	3	3	30	44	8	52	0
3	5	4	5	4	31	44	8	52	0
4	8	6	8	6	32	44	8	52	0
5	10	7	12	5	33	43	8	51	0
6	11	8	14	5	34	42	8	50	0
7	11	8	15	4	35	41	8	49	0
8	13	8	18	3	36	40	8	48	0
9	18	8	23	3	37	39	8	47	0
10	21	8	26	3	38	38	8	46	0
11	24	8	29	3	39	34	8	42	0
12	26	8	32	2	40	31	8	39	0
13	28	8	34	2	41	30	8	38	0
14	30	8	36	2	42	28	8	36	0
15	31	8	37	2	43	26	8	34	0
16	34	8	41	1	44	24	8	32	0
17	38	8	46	0	45	21	8	29	0
18	39	8	47	0	46	18	8	26	0
19	40	8	48	0	47	13	8	21	0
20	41	8	49	0	48	11	8	19	0
21	42	8	50	0	49	11	8	19	0
22	43	8	51	0	50	10	7	17	0
23	44	8	52	0	51	8	6	14	0
24	44	8	52	0	52	5	4	9	0
25	44	8	52	0	53	3	3	6	0
26	44	8	52	0	54	2	2	4	0
27	44	8	52	0	55	1	1	2	0

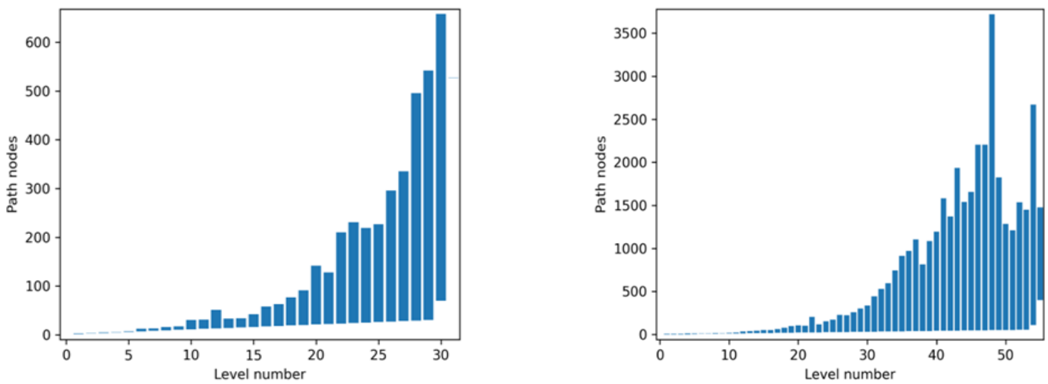
When the research expands and generate the path nodes for each node in the domain to goal node. The Tables 9 and 10 shows the paths range for each node at the different levels. Figure 5 presents the relation between level number (Figure 5(a)) and path nodes (Figure 5(b)) for Tables 9 and 10 consequently.

Table 9. Path nodes range count for each node in slide tile puzzle 3×3

Level	Path nodes range	Level	Path nodes range	Level	Path nodes range	Level	Path nodes range
1	2	9	10-17	17	18-63	25	26-227
2	3	10	11-30	18	19-77	26	27-296
3	4	11	12-31	19	20-91	27	28-335
4	5	12	13-51	20	21-142	28	29-496
5	6	13	14-33	21	22-128	29	30-542
6	7-12	14	15-34	22	23-210	30	70-658
7	8-13	15	16-42	23	24-231	31	527
8	9-16	16	17-58	24	25-219		

Table 10. Path nodes range count for each node in slide tile puzzle 2×5

Level	Path nodes range	Level	Path nodes range	Level	Path nodes range	Level	Path nodes range
1	2	15	16-51	29	30-302	43	44-1,934
2	3	16	17-52	30	31-336	44	45-1,541
3	4	17	18-64	31	32-442	45	46-1,658
4	5	18	19-77	32	33-530	46	47-2,206
5	6	19	20-96	33	34-596	47	48-2,207
6	7-12	20	21-105	34	35-744	48	49-3,720
7	8-13	21	22-101	35	36-912	49	50-1,824
8	9-14	22	23-205	36	37-972	50	51-1,286
9	10-15	23	24-115	37	38-1,106	51	52-1,210
10	11-20	24	25-152	38	39-816	52	53-1,537
11	12-22	25	26-172	39	40-1,084	53	54-1,448
12	13-36	26	27-228	40	41-1,194	54	108-2,672
13	14-37	27	28-224	41	42-1,582	55	398-1,478
14	15-42	28	29-258	42	43-1,372		



(a) (b)

Figure 5. Relation between (a) level number and (b) path nodes

When extract form the total domain of the two cases study nodes similar in empty node location compare to goal states. The node count for each empty location is similar and equal for the number of total domain nodes divide by locations count in slide tile puzzle 181,440/9=2,060, 181,440/10=181,440). The level range is huge and also MD range which fall in approximately between low and high boundary. Tables 11 and 12 present a comparison between empty tile locations and level range, MD range, and number of nodes, where Table 11 shows a slide tile puzzle 3×3 and Table 12 shows a slide tile puzzle 2×5.

Table 11. Empty tile location compares to level range, MD range and nodes count with slide tile puzzle 3×3

Tile value	Levels range	MD range	Value count
Empty	4-30	8-24	20,160
1	0-30	0-20	20,160
2	5-30	4-22	20,160
3	9-30	4-22	20,160
4	5-30	4-22	20,160
5	7-30	4-24	20,160
6	9-31	6-24	20,160
7	9-30	4-22	20,160
8	9-31	6-24	20,160

Table 12. Empty tile location compares to level range, MD range and nodes count with slide tile puzzle 2×5

Tile value	Levels range	MD range	Value count
Empty	5-55	10-34	181,440
1	0-47	0-28	181,440
2	6-50	4-30	181,440
3	10-52	4-32	181,440
4	14-54	6-32	181,440
5	18-54	8-32	181,440
6	6-48	4-30	181,440
7	8-50	4-32	181,440
8	12-53	6-34	181,440
9	16-54	8-34	181,440

4. CONCLUSION

This research concentrated on the calculated value of MD, domain count, level range, and empty tile location. The results of MD weaknesses are related to the spread range of MD in each level in the slide tile puzzle. The detailed reasons for low bound value and other related reasons for its weakness are discussed in this work. This paper approved that low bound value is not the critical issue that complicates the search domain but is related to the distribution of MD value in the domain. The summation way used in MD for all tiles affects the duplication values calculated. The total node in the optimal path also affects the search performance. The recommendations for future work are to apply these results at depth-first search with an iterative model and check expected results on other domains. Also, Advanced heuristics like pattern databases and WD can be combined with MD to improve the accuracy of estimating the cost of achieving the goal state in slide tile puzzles.

FUNDING INFORMATION

This research is funded by the Deanship of Scientific Research in Zarqa University, Jordan.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Mohammed Nayef Al Refai	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓
Zeyad Mohammed Jamhawi	✓	✓				✓		✓	✓	✓	✓	✓		
Ahmed Ali Otoom	✓		✓	✓			✓		✓	✓	✓		✓	
Adai AL-Momani	✓	✓	✓		✓		✓			✓	✓			
Hayel Khafajeh	✓		✓		✓		✓			✓			✓	
Issa Atoum	✓	✓	✓			✓	✓			✓				

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Derived data supporting the findings of this study are available from the corresponding author [AMN] on request.

REFERENCES

[1] R. Suwanda, Z. Syahputra, and E. M. Zamzami, "Analysis of Euclidean distance and Manhattan distance in the k-means algorithm for variations number of centroid k," *Journal of Physics: Conference Series*, vol. 1566, no. 1, 2020, doi: 10.1088/1742-6596/1566/1/012058.

[2] J. T. Temple, "Characteristics of distance matrices based on Euclidean, Manhattan and Hausdorff coefficients," *Journal of Classification*, vol. 40, no. 2, pp. 214–232, 2023, doi: 10.1007/s00357-023-09435-1.

[3] M. Faisal, E. M. Zamzami, and Sutarman, "Comparative analysis of inter-centroid k-means performance using Euclidean distance, Canberra distance and Manhattan distance," *Journal of Physics: Conference Series*, vol. 1566, no. 1, 2020, doi: 10.1088/1742-6596/1566/1/012112.

[4] D. E. Martin and D. E. Martin, "Models of computational systems-cyclic to acyclic graph transformations," *IEEE Transactions on Electronic Computers*, no. 1, pp. 70–79, 1967, doi: 10.1109/PGEC.1967.264607.

[5] Z. Wu, T. Song, and Y. Zhang, "Quantum k-means algorithm based on Manhattan distance," *Quantum Information Processing*, vol. 21, no. 1, 2022, doi: 10.1007/s11128-021-03384-7.

[6] G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, A. Nannarelli, M. Re, and S. Spanò, "N-dimensional approximation of Euclidean distance," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 3, pp. 565-569, 2020, doi: 10.1109/TCSII.2019.2919545.

[7] W. Wang, S. Li, and Y. Sun, "Application of a novel improved Manhattan distance on bearing fault diagnosis," *Research Square*, pp. 1-22, Mar. 2023, doi: 10.21203/rs.3.rs-2599124/v1.

[8] T. Liu and D. Tao, "On the performance of Manhattan nonnegative matrix factorization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 9, pp. 1851–1863, 2016, doi: 10.1109/TNNLS.2015.2458986.

[9] M. D. Malkauthekar, "Analysis of Euclidean distance and Manhattan distance measure in face recognition," *IET Conference Publications*, vol. 2013, pp. 503–507, 2013, doi: 10.1049/cp.2013.2636.

[10] X. Jing and X. Yang, "Application and improvement of heuristic function in A ∗ Algorithm," *Chinese Control Conference, CCC*, pp. 2191–2194, 2018, doi: 10.23919/ChiCC.2018.8482630.

[11] X. Xu, Z. Zeng, and Y. Chang, "A multidimensional journal evaluation framework based on the Pareto-dominated set measured by the Manhattan distance," *Learned Publishing*, vol. 36, no. 4, pp. 619–637, 2023, doi: 10.1002/leap.1571.

[12] Y. C. Wang, Y. Xing, and J. Zhang, "Voronoi treemap in Manhattan distance and Chebyshev distance," *Information Visualization*, vol. 22, no. 3, pp. 246–264, 2023, doi: 10.1177/14738716231167181.

[13] J. T. Thayer, W. Ruml, and J. Kreis, "Using distance estimates in heuristic search: a re-evaluation," *International Symposium on Combinatorial Search, SoCS 2009*, vol. 19, no. 1, pp. 382–385, 2009.

[14] R. P. Kondekar *et al.*, "Implementation and analysis of iterative MapReduce based heuristic algorithm for solving N-puzzle," *Journal of Computers*, vol. 9, no. 2, 2014, doi: 10.4304/jcp.9.2.420-424.

[15] R. E. Korf and A. Felner, "Disjoint pattern database heuristics," *Artificial Intelligence*, vol. 134, no. 1–2, pp. 9–22, 2002, doi: 10.1016/S0004-3702(01)00092-3.

[16] R. Natarajan, M. Saleem, S. Aine, M. Likhachev, and H. Choset, "A-MHA*: anytime multi-heuristic A*," *Proceedings of the International Symposium on Combinatorial Search*, vol. 10, no. 1, pp. 192–193, 2021, doi: 10.1609/socs.v10i1.18486.

[17] A. Halme, "Cooperative heuristic search with software agents," *M.Sc. Thesis*, School of Science, Aalto University, Espoo, Finland, 2014.

[18] A. Felner, U. Zahavi, R. Holte, J. Schaeffer, N. Sturtevant, and Z. Zhang, "Inconsistent heuristics in theory and practice," *Artificial Intelligence*, vol. 175, no. 9–10, pp. 1570–1603, 2011, doi: 10.1016/j.artint.2011.02.001.

[19] Y. -Q. Wang, "An analysis of the Viola-Jones face detection algorithm," *Image Processing On Line*, vol. 4, pp. 128–148, 2014, doi: 128-148. 10.5201/ipol.2014.104.

[20] M. N. Al-Refai and Z. M. Jamhawi, "Empirical study prove that breadth-first search is more effective memory usage than depth-first search in frontier boundary cyclic graph," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 2, pp. 265–272, 2021, doi: 10.11591/ijai.v10.i2.pp265-272.

[21] D. O. Hasan, A. M. Aladdin, H. S. Talabani, T. A. Rashid, and S. Mirjalili, "The fifteen puzzle-a new approach through hybridizing three heuristics methods," *Computers*, vol. 12, no. 1, 2023, doi: 10.3390/computers12010011.

[22] Y. F. Yiu, J. Du, and R. Mahapatra, "Evolutionary heuristic A∗ search: heuristic function optimization via genetic algorithm," *Proceedings - 2018 1st IEEE International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2018*, pp. 25–32, 2018, doi: 10.1109/AIKE.2018.00012.

[23] S. S. Mim, M. N. Islam, and D. Logofatu, "Examining various search algorithms in AI with appropriate literature and their performances against a human agent in the snake game," *IEEE Global Engineering Education Conference, EDUCON*, 2023, doi: 10.1109/EDUCON54358.2023.10125253.

[24] A. Auer and H. Kaindl, "A case study of revisiting best-first vs. depth-first search," *Frontiers in Artificial Intelligence and Applications*, vol. 110, pp. 141–145, 2004.

[25] Y. Zeng, H. Ren, T. Yang, S. Xiao, and N. Xiong, "A novel similarity measure of single-valued neutrosophic sets based on modified Manhattan distance and its applications," *Electronics*, vol. 11, no. 6, Mar. 2022, doi: 10.3390/electronics11060941.

[26] S. Ramakrishnan, L. Sevalaiappan, and S. Ravichandran, "Traffic-aware clustering scheme for manet using modified elephant herding optimization algorithm," *International Arab Journal of Information Technology*, vol. 18, no. 5, pp. 694–703, 2021, doi: 10.34028/iajit/18/5/9.

BIOGRAPHIES OF AUTHORS

Dr. Mohammed Nayef Al Refai is vice dean of Higher Studies in Zarqa University Ph.D. in computer science (distributed systems) Amman Arab University for graduate studies 2007, master degree in computer science, Al Albeit University Jordan, 1999-2002. B.S. in computer science, Mu'ta University, Alkarak, Jordan, 1988-92. He can be contacted at email: refai@zu.edu.jo.

Dr. Zeyad Mohammed Jamhawi is Ph.D. in computer information systems (artificial intelligence), Omdurman Islamic University, Sudan 2013-2016. Master degree in computer information system, Arab Academy for Banking and Finance, Amman, Jordan 2009-2012. Professional diploma specialization in E-government, Jordanian University, Jordan 2007-2008. B.S. in computer science, Mu'ta University, Alkarak, Jordan, 1988-92. He can be contacted at email: z_jamhawi@asu.edu.jo.

Dr. Ahmed Ali Otoom is chairman of Department of Cybersecurity and Cloud Computing in Applied Science Private University, Ph.D. in computer science (distributed systems) Amman Arab University for graduate studies 2007, master degree in computer science, USA. B.S. in computer science, Mu'ta University, Alkarak, Jordan, 1988-92. He can be contacted at email: a_otoom@asu.edu.jo.

Adai Al-Momani received the B.Sc. degrees in computer science from Jadara University, Jordan, M.Sc. degrees in information technology from University Tenaga Nasional, Malaysia, in 2010 and 2016, respectively and the Ph.D. degree in computer sciences from Universiti Sultan Zainal Abidin, Malaysia in 2023. His research interests include internet of things and computer information system. He can be contacted at email: oalmomani@zu.edu.jo.

Assoc. Prof. Hayel Khafajeh obtained his Ph.D. in computer information systems in 2008 in Jordan. He joined Zarqa University, Jordan in 2009. In 2010, he served as head of the CIS Department for two years. Since academic year 2014/2015, he has served and he still as the vice dean of the IT College at Zarqa. He has worked for 23 years in the educational field as programmer, teacher's supervisor, head of IT division, and manager of ICDL Center. He has published many educational computer books for the Ministry of Education in Jordan. In 2013, he published his Introduction to Java Programming book for university students. His research interests include information retrieval, AI, and E-learning. He is the author of several publications on these topics. He can be contacted at email: hayelkh@zu.edu.jo.

Dr. Issa Atoum is member in Faculty of Information Technology, Philadelphia University, Ph.D. in software engineering, Unimas University Malaysia, 2015. Master degree in computer information system, Philadelphia University, Jordan 2012. B.S. in computer science, Yarmouk University, Jordan, 1997. He can be contacted at email: iatoum@Philadelphia.edu.jo.