Solving the $(n^2 - 1)$-Puzzle in Real Time with $\frac{8}{3}n^3$ Expected Moves

Ian Parberry

Technical Report LARC-2014-01

Laboratory for Recreational Computing
Department of Computer Science & Engineering
University of North Texas
Denton, Texas, USA

April, 2014

Solving the (n^2-1)-Puzzle in Real Time with $\frac{8}{3}n^3$ Expected Moves

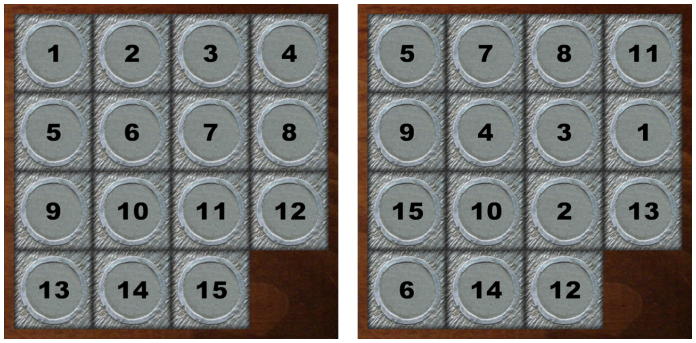
Ian Parberry*
Department of Computer Science & Engineering
University of North Texas

Abstract

It is shown theoretically and verified experimentally that a solution for the (n^2-1)-puzzle with expected number of moves $\frac{8}{3}n^3 + O(n^2)$ can be found in real time.

1 Introduction

The (n^2-1)-puzzle is defined as follows. Given n^2-1 numbered tiles arranged in row-major order in an $n \times n$ grid leaving a blank space in the last position where one tile is missing, the aim is to scramble the puzzle and return it to the initial conguration by repeatedly sliding an adjacent tile into the blank location. For example, the following image shows an example for $n = 4$ of the solved configuration (left) and a random configuration (right).



The 15-puzzle has a long and interesting history (see, for example, Hordern [2]) that is said to date back to the 1870s. More recently, the 15-puzzle has appeared in the form of various apps on mobile devices and as minigames inside larger games. For example, the 15-puzzle can be found in the original *Final Fantasy* (Square Enix, 1987) and *The Legend of Zelda: The Windwaker* (Nintendo 2003), and the 8-puzzle can be found in *Machinarium* (Amanita Design, 2009).

Ratner and Warmuth [6] have proved that the problem of finding the minimum number of moves for the (n^2-1)-puzzle is NP-hard, and they demonstrate that a polynomial time approximation algorithm exists. Kornhauser, Miller, and Spirakis [3] show an $O(n^3)$ time algorithm for the (n^2-1)-puzzle, which therefore uses $O(n^3)$ moves in the worst case. Parberry [5] gave worst case upper and

*Author's address: Dept. of Computer Science & Engineering, Univ. of North Texas, 1155 Union Circle #311366, Denton, Texas 76203–5017, U.S.A. URL: http://larc.unt.edu/ian.

lower bounds of $5n^3$ and n^3, respectively, on the number of moves required to solve the (n^2-1)-puzzle. The algorithm in the latter paper is *real time*, that is, the first move can be made in time $O(1)$ after the algorithm begins, whereas the other algorithms appear to require $\Omega(n^3)$ time before the first move can be made.

Whilst upper bounds are certainly interesting, a hypothetical player faced with solving a random configuration of the puzzle is likely be more concerned about the expected number of moves than the worst case. With that in mind, Parberry [5] gave lower bounds of at least $2n^3/3$ for the expected number of moves, and at least $0.264n^3$ moves for a random configuration with probability 1. We extend this work by showing both theoretically and experimentally that the real-time algorithm from [5] solves the (n^2-1)-puzzle in expected number of moves $\frac{8}{3}n^3 + O(n^2)$.

2 The Algorithm

The real-time algorithm from Parberry [5] works as follows on an $n \times n$ instance of the puzzle. We will refer to the grid position in row i and column j, for $0 \leq i, j < n$ as *position*(i, j), and refer to the tile that belongs there as *tile* (i, j). There are sequences of 5 moves that bring a tile one place horizontally or vertically, that is, from position (i, j) to positions $(i-1, j)$, $(i+1, j)$, $(i, j-1)$, or $(i, j+1)$, and a sequence of 6 moves that brings a tile one place diagonally, that is, from position (i, j) to positions $(i-1, j-1)$, or $(i+1, j+1)$. Move the blank to the position immediately above tile $(0, 0)$, then use a sequence of these moves to bring that tile to position $(0, 0)$. Repeat this for tiles $(0, 1), (0, 2), \ldots, (0, n-1)$, taking care not to disturb the work that has already been done. Tiles $(0, n-2)$ and $(0, n-1)$ require a few extra moves to flip into place. Once the first row has been completed, do likewise for the first column. Once the first row and column are in place, recurse on the remaining $(n-1) \times (n-1)$ puzzle. The base of the recursion is $n = 3$, which can be solved by brute force in no more than 31 moves (Reinefeld [7]).

3 Theoretical Analysis

Suppose n is even (the case where k is odd is left as an exercise for the interested reader). Consider the expected number of moves required to solve the first half of the first row of the puzzle. For each of those $n/2$ tiles $t \in \{(0, k) | 0 \leq k < n/2)\}$, the expected number of moves required to move tile $t = (i, k)$ to position (i, k) will be equal to the sum over all positions p of the number of moves required to move t from position p to position (i, k), divided by n^2. We will do this in two parts, the the number of moves required to move the blank into place above tile t (Section 3.1), and the number of moves required to move t to position (i, k) (Section 3.2) once the blank is in place.

3.1 The Blank

The sum of the number of moves required to move the blank from position $(0, 0)$ to each of the n^2 possible destinations is:

$$\sum_{i=0}^{n-1} \sum_{j=i}^{i+n-1} j = n^3 - n^2.$$

0	1	2	3	4	5	6	7
1	2	3	4	5	6	7	8
2	3	4	5	6	7	8	9
3	4	5	6	7	8	9	10
4	5	6	7	8	9	10	11
5	6	7	8	9	10	11	12
6	7	8	9	10	11	12	13
7	8	9	10	11	12	13	14

1	0	1	2	3	4	5	6
2	1	2	3	4	5	6	7
3	2	3	4	5	6	7	8
4	3	4	5	6	7	8	9
5	4	5	6	7	8	9	10
6	5	6	7	8	9	10	11
7	6	7	8	9	10	11	12
8	7	8	9	10	11	12	13

2	1	0	1	2	3	4	5
3	2	1	2	3	4	5	6
4	3	2	3	4	5	6	7
5	4	3	4	5	6	7	8
6	5	4	5	6	7	8	9
7	6	5	6	7	8	9	10
8	7	6	7	8	9	10	11
9	8	7	8	9	10	11	12

3	2	1	0	1	2	3	4
4	3	2	1	2	3	4	5
5	4	3	2	3	4	5	6
6	5	4	3	4	5	6	7
7	6	5	4	5	6	7	8
8	7	6	5	6	7	8	9
9	8	7	6	7	8	9	10
10	9	8	7	8	9	10	11

Figure 1: Number of moves required to move the blank to each position from the first half of the first row of the 63-puzzle.

For example, with $n = 8$, looking at Figure 1 (left) and numbering the rows from 0 to $n-1$ top-to-bottom, row i has sum $\sum_{j=i}^{i+n-1} j$. For position $(0,k)$, $0 \le k < n/2$, the sum is:

$$\begin{aligned} & \left(n^3 - n^2\right) - n^2 k + 2n \sum_{i=1}^{k} i \\ = \; & \left(n^3 - n^2\right) - n(n-1)k + nk^2. \end{aligned}$$

(See, for example, Figure 1 with $n = 8$ and $k = 0, 1, 2, 3$ from left to right.) Therefore, the total number of moves for positioning the blank above tiles $\{(0,k) | 0 \le k < n/2)\}$ is:

$$\begin{aligned} & \frac{n}{2}\left(n^3 - n^2\right) - n(n-1) \sum_{k=1}^{n/2} k + n \sum_{k=1}^{n/2} k^2 \\ = \; & \frac{5}{12} n^4 - \frac{1}{4} n^3 - \frac{1}{6} n^2. \end{aligned}$$

Therefore total number of moves for moving the blank into place while solving the first row and the first column is less than:

$$\begin{aligned} & 4\left(\frac{5}{12} n^4 - \frac{1}{4} n^3 - \frac{1}{6} n^2\right) - \left(n^3 - n^2\right) \\ = \; & \frac{1}{3}\left(5n^4 - 6n^3 + n^2\right). \end{aligned}$$

3.2 The Tiles

The sum of the number of moves required to move tile $(0,0)$ to position $(0,0)$ from all of the n^2 possible sources, is (see Figure 2, left):

$$\begin{aligned} & 2\left(5 \sum_{i=1}^{n-1} i^2 + \sum_{i=1}^{n-2} \sum_{j=1}^{i} j\right) + 6 \sum_{i=1}^{n-1} i \\ = \; & \frac{11}{3} n^3 - 2n^2 - \frac{2}{3} n \end{aligned}$$

0	5	10	15	20	25	30	35
5	6	11	16	21	26	31	36
10	11	12	17	22	27	32	37
15	16	17	18	23	28	33	38
20	21	22	23	24	29	34	39
25	26	27	28	29	30	35	40
30	31	32	33	34	35	36	41
35	36	37	38	39	40	41	42

5	0	5	10	15	20	25	30
6	5	6	11	16	21	26	31
11	10	11	12	17	22	27	32
16	15	16	17	18	23	28	33
21	20	21	22	23	24	29	34
26	25	26	27	28	29	30	35
31	30	31	32	33	34	35	36
36	35	36	37	38	39	40	41

10	5	0	5	10	15	20	25
11	6	5	6	11	16	21	26
12	11	10	11	12	17	22	27
17	16	15	16	17	18	23	28
22	21	20	21	22	23	24	29
27	26	25	26	27	28	29	30
32	31	30	31	32	33	34	35
37	36	35	36	37	38	39	40

15	10	5	0	5	10	15	20
16	11	6	5	6	11	16	21
17	12	11	10	11	12	17	22
18	17	16	15	16	17	18	23
23	22	21	20	21	22	23	24
28	27	26	25	26	27	28	29
33	32	31	30	31	32	33	34
38	37	36	35	36	37	38	39

Figure 2: Number of moves required to move a tile from each position to the first half of the first row of the 63-puzzle.

For position $(0, k)$, $0 \leq k < n/2$, the sum is (see Figure 2):

$$\begin{aligned} & \left(\frac{11}{3}n^3 - 2n^2 - \frac{2}{3}n\right) - (3n^2 - 4n - 4)k + (6n + 4)\sum_{i=1}^{k-1} i \\ = & \left(\frac{11}{3}n^3 - 2n^2 - \frac{2}{3}n\right) - (3n^2 - 7n - 6)k + (3n + 2)k^2. \end{aligned}$$

Therefore, the total number of moves for moving tiles $\{(0, k)|0 \leq k < n/2)\}$ into place in the first half of the first row is:

$$\begin{aligned} & \frac{n}{2}\left(\frac{11}{3}n^3 - 2n^2 - \frac{2}{3}n\right) - (3n^2 - 7n - 6)\sum_{k=1}^{n/2} k + (3n + 2)\sum_{k=1}^{n/2} k^2 \\ = & \frac{19}{12}n^4 - \frac{11}{12}n^3 - \frac{1}{3}n^2 - \frac{1}{2}n, \end{aligned}$$

and the total number of moves for moving the first row and column tiles into place is at most:

$$\begin{aligned} & 4\left(\frac{19}{12}n^4 - \frac{11}{12}n^3 - \frac{1}{3}n^2 - \frac{1}{2}n\right) - \left(\frac{11}{3}n^3 - 2n^2 - \frac{2}{3}n\right) \\ = & \frac{1}{3}\left(19n^4 - 22n^3 + 5n^2 - 2n\right). \end{aligned}$$

The astute reader will have noticed that we have under-counted by $O(n)$ to bring the blank into position at the start, and $O(1)$ for the last tile in the row and column. This is more than compensated for by the fact that we have over-counted by $O(n^3)$ when moving the blank in Section 3.1 since there is never any need to move the blank to the last row.

3.3 Tiles and Blank Together

The total number of moves used to solve the first row and column is at most the sum of the results of Sections 3.1 and 3.2. That is,

$$\begin{aligned} & \frac{1}{3}\left(19n^4 - 22n^3 + 5n^2 - 2n\right) + \frac{1}{3}\left(5n^4 - 6n^3 + n^2\right) \\ = & 8n^4 - \frac{28}{3}n^3 + 2n^2 - \frac{2}{3}n. \end{aligned}$$

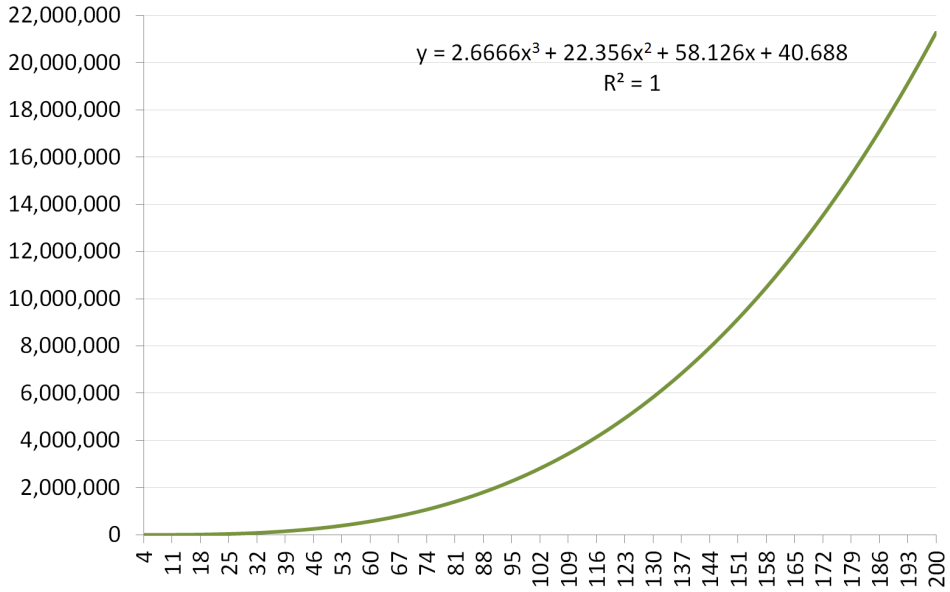


Figure 3: The average number of moves required to solve 10,000 random instances of the $(n^2 - 1)$-puzzle for $4 \leq n \leq 200$.

The expected number of moves to solve the first row and column is therefore $8n^2 + O(n)$, and so the expected number of moves to solve the whole puzzle is bounded above by:

$$8 \sum_{i=2}^{n} i^2 + O(n^2) = \frac{8}{3} n^3 + O(n^2).$$

4 Experimental Analysis

We generated 10,000 random instances of the $(n^2 - 1)$-puzzle for all n such that $4 \leq n \leq 200$ using the standard algorithm for generating a random even permutation based on the Mersenne Twister (Matsumoto and Nishimura [4]) seeded with the number of milliseconds since system reboot. We then solved each instance using the algorithm of Parberry [5] and measured the average number of moves required to solve each size, which should approximate the expected value if the sample size is large enough. We found that the average number of moves tends to $2.6666n^3 + O(n)$ with an R^2 value of 1.000 (see Figure 3). In fact, the number of moves divided by n^3 is less than 2.66 for $4 \leq n \leq 200$ (see Figure 4).

It is perhaps interesting to note that while the maximum number of moves required to solve the 15-puzzle is 80 (Brüngger et al. [1]), in 10^9 random trials the algorithm of Parberry [5] used expected number of moves 126.73 and a maximum of 217 moves.

5 Conclusion and Open Problems

We have shown both theoretically and experimentally that the real-time algorithm from [5] solves the $(n^2 - 1)$-puzzle in expected number of moves $\frac{8}{3}n^3 + O(n^2)$. However, the best known lower

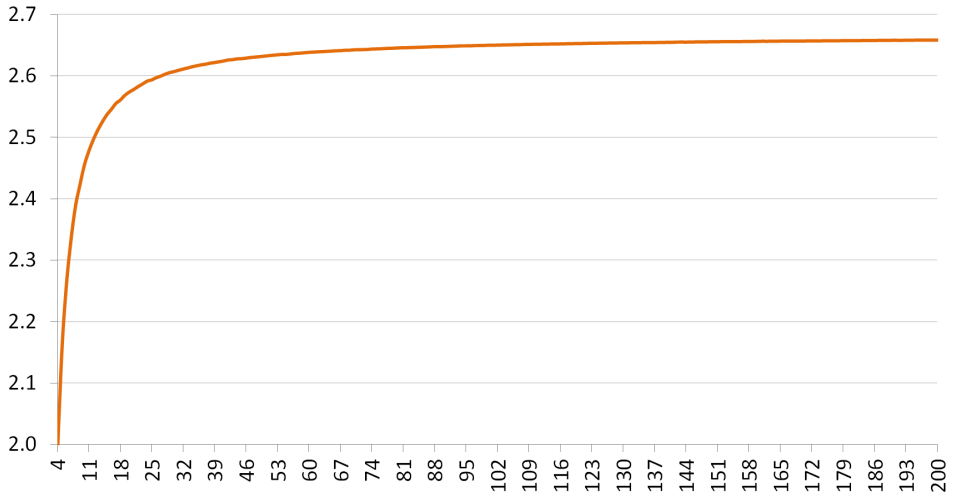


Figure 4: The average number of moves required to solve 10,000 random instances of the $(n^2 - 1)$-puzzle divided by n^3 for $4 \leq n \leq 200$.

bound for the expected number of moves is $2n^3/3$ from Parberry [5]. We conjecture that there is almost certainly an algorithm with a smaller expected number of moves, and that the lower bound likewise is almost certainly not tight.

References

[1] Adrian Brüngger, Ambros Marzetta, Komei Fukuda, and Jurg Nievergelt. The parallel search bench ZRAM and its applications. *Annals of Operations Research*, 90:45–63, 1999.

[2] L. E. Hordern. *Sliding Piece Puzzles*. Oxford University Press, 1986.

[3] Daniel Martin Kornhauser, Gary Miller, and Paul Spirakis. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. In *Proceedings of the 25th Annual Symposium on Foundations of Computer Science*, pages 241–250, 1984.

[4] Makoto Matsumoto and Takuji Nishimura. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, 1998.

[5] Ian Parberry. A real-time algorithm for the $(n^2 - 1)$-puzzle. *Information Processing Letters*, 56:23–28, 1995.

[6] Daniel Ratner and Manfred Warmuth. The $(n^2 - 1)$-puzzle and related relocation problems. *Journal of Symbolic Computation*, 10(2):111–137, July 1990.

[7] Alexander Reinefeld. Complete solution of the eight-puzzle and the benefit of node ordering in IDA*. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 248–253, 1993.