Domino Tiling an N-Queens Chessboard

Christopher Sakaliyski*

Advisor
Gene Li†

November 30, 2022

Abstract

In this paper, we combine the n-queens and the domino tiling problems. To do so, we calculate the number of domino tiling arrangements of the solutions to the n-queens problem. Based on results presented for $n \in [2, 15]$, we conjecture that having an equal number of black and white tiles remaining after placing the queens is sufficient for tilability, whereas this is not always true for other mutilated chessboards, as the mutilations could split the board in small disjoint groups. Our numerical calculations also suggest that tilability sometimes occurs when $n \equiv 0, 1 \pmod 4$. On the other hand, for $n \equiv 2, 3 \pmod 4$, tilability never occurs.

*Whitney M. Young Magnet High School
†Toyota Technological Institute

Contents

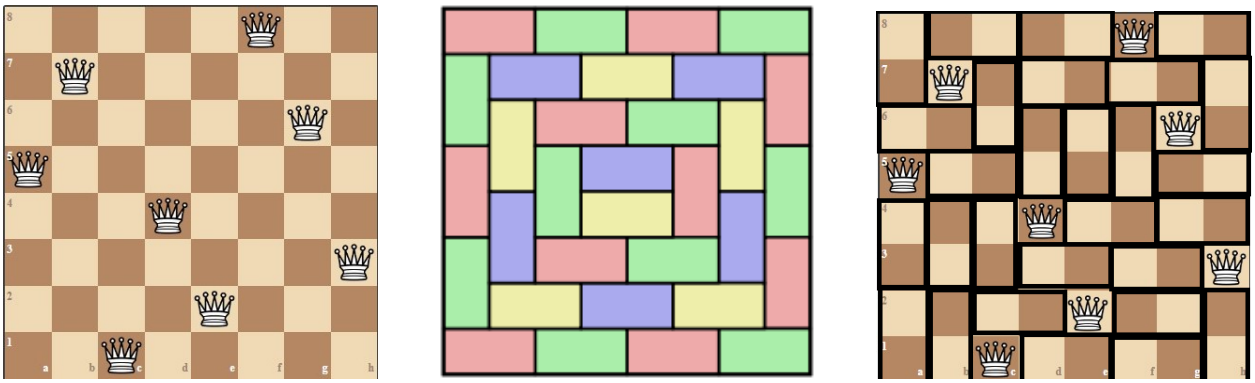
1 Introduction

In this paper, we study the problem of domino tiling the solutions to the n-queens problem.

In the n-queens problem, n queens are placed on an n by n chessboard such that no queens are attacking each other. This means that no two queens are on the same row, column, or diagonal. A possible solution to the 8-queens problem is shown in Figure 1a.

In the domino tiling problem, 1x2 tiles are placed on an n by n chessboard such that they are non-overlapping and fill up the board. A solution to tiling an 8 by 8 board is shown in Figure 1b.

For every solution to the n-queens problem, we check if it is tileable and only for $n \in [2, 11]$ we calculate the number of domino tilings for that solution. For $n \in [12, 15]$, the number of tilings for each n-queens solution is already large enough so we only check for tilability.



(a) No three queens in a line. **(b)** Domino tiling[Wik22]. **(c)** Domino tiling of 1a

Figure 1: Example of an 8-queens solution, tiling an 8x8 board and tiling an 8-queens solution.

1.1 Our Contributions

We devise and implement efficient algorithms which compute all solutions to the n-queens problem and calculate the number of domino tilings for a given solution, all for $n \in [2, 15]$. Based on the numerical results our program has given us, we conjecture the following:

- An n-queens chessboard can be tiled if the number of white queens and black queens is equivalent. This is interesting as this condition is not sufficient for all mutilated chessboards.
- Some of n-queens solutions can be tiled for $n \equiv 0, 1 \pmod 4$ and none can be tiled for $n \equiv 2, 3 \pmod 4$.

1.2 Related Work

We discuss related work to the two problems in this subsection.

1.2.1 N-Queens Problem

The problem for $n = 8$ was first introduced by the chess composer Max Bazzel in 1848 [Bal60]. Two years later, in 1850, Franz Nauck published his solutions in the Leipziger Illustrierte Zeitung and expanded the problem by changing it to n-queens [EST92]. Many famous mathematicians such as Gauss worked on this problem.

Along with regular solutions, the fundamental solutions are all the solutions which are not rotations or reflections of each other, which gives about $\frac{1}{8}$ as many fundamental solutions. This is due to there being four rotations (90°, 180°, 270°, 360° degrees clockwise) and four reflections (about the horizontal axis, the vertical axis, and both diagonals) of the placements [EST92]. The number of fundamental solutions of the 8-queens problem has been proved to be only 12 by English mathematician J. W. L. Glaisher [EST92]. There exists an algorithm for finding only the fundamental solutions. Rodney Topor optimized and simplified the algorithm presented by Fillmore and Williamson by using orbits and groups to generate only the fundamental solutions [Top82].

In 2021, Michael Simkin proved that for large numbers of n, the number of solutions is approximately $(0.143n)^n$, which grows asymptotically [Sim21]. More specifically, the formula, which will not be used in this paper, is

$$\mathcal{Q}(n) = ((1 \pm o(1))ne^{-\alpha})^n.$$

1.2.2 Domino Tiling Problem

The domino tiling problem originates from Tatami, which are Japanese floor mats in the shape of a 1x2 rectangle. These mats are used to tile rooms, but with a bit more restriction on the tiling. An example of this is shown in Figure 1b.

The formula for domino tiling arrangements for board m by n was discovered independently by Temperley and Fischer [TF61], and Kasteleyn [Kas61], both in 1961:

$$\prod_{j=1}^{\lceil \frac{m}{2} \rceil} \prod_{j=1}^{\lceil \frac{n}{2} \rceil} \left(4\cos^2 \frac{\pi j}{m+1} + 4\cos^2 \frac{\pi k}{n+1} \right).$$

2 Problem Setup

A solution to the n-queens problem is an arrangement of the n queens which meets the non-attacking placement constraints.

We say that a chessboard is mutilated if some squares are left out. Mutilations are the empty squares on the chessboard. Mutilated chessboards are tileable if they can be tiled in at least one way. We know that a chessboard is made up of alternating black and white squares. This fact can be used to see that every tile covers one black and one white square. Therefore, a mutilated chessboard can never be tiled if the number of black and white squares are nonequivalent. The contrapositive of this statement is not always true, as empty squares could split the board into untileable groups. In the case of n-queens however, the empty squares, or the squares on which the queens are placed, are spread far enough apart so that the whole board is connected as a group.

3 Programs

In this section we discuss the programs which solve the two problems and combine them.

3.1 N-Queens Program

Brute forcing all $\binom{n^2}{n} \approx n^{2n}$ placements of queens with $O(n^{2n})$ runtime is wildly inefficient. Using a brute-force approach on larger n will take too much time to compute all solutions with this runtime.

Instead of brute force, the method used in this program is called backtracking. Backtracking is used to search for all possibilities one step at a time. It checks the constraints every time, and if the constraints are not met, it backtracks and goes one step back. If the constraints are met, then it goes to the next step. This is significantly faster, as the steps for the n-queens algorithm just place one queen in every column. This gives a runtime of $O(n!) \approx O(n^n)$. This is much faster than the $O(n^{2n})$ runtime used in brute force.

Our constraints are that no two queens are on the same row, column, or diagonal. The program starts at the first column of the chessboard. At every column, the program tries to place a queen in that column. For every square in that column, the program checks if the queen can be placed with the constraints being followed. This means that the new queen is not on the same row or diagonal as previous queens. Columns do not need to be checked due to every step being on different columns. If the queen can be placed in that column, then it is placed and the program moves onto the next column. After the program is done going through the rest of the columns, it goes back and removes the queen it placed. The program does the same thing for the next square in the column, and so on.

Algorithm 1: Pseudo code for n-queens program.

```
placeQueens(current column)
{
    if current column is n
    {
        add one to counter of solutions
        break
    }
    for all squares in the current column
    {
        check if the square is attacked by any queens in the same row
        check if the square is attacked by any queens on both diagonals
        if the square is not attacked
        {
            place the queen
            placeQueens(next column)
            remove the queen
        }
    }
}
```

3.2 Domino Tiling Program

In the domino tiling program, instead of repeatedly trying to place tiles by brute-forcing, the program goes step by step and backtracks. The program goes through the rows, starting from the top left on the top most row and ending at the bottom right on the bottom row. There are two types of tiles that can be placed, tiles that go left-right, and tiles that go up-down.

The program starts at the first square. On the current square, it checks if a tile can be placed such that it covers the current square and the square to the right. If it can, it places the tile and goes on to the next square with this same process. If it cannot, it checks if a tile can be placed such that it covers the current square and the square below. It then goes on to the next square with the same process. A boolean array is used to know whether a square is already covered by a tile or not, and it can be toggled when tiles get placed. A tile always needs to be placed on the current square because later steps cannot place a tile above or to the left of the square they are on. The program ends when the next square it lands on is not on the chessboard as that is when every square is filled.

Algorithm 2: Pseudo code for domino tiling program.

```
placeTiles(current x, current y)
{
    if all tiles have been placed
    {
        add one to counter of solutions
        break
    }
```

```
    if the current square has been covered
    {
        placeTiles(next x, next y)
    }
    set the tile to be covered in the boolean array
    if it is possible to place a left-right tile
    {
        place the left-right tile
        placeTiles(next x, next y)
        remove the tile
    }
    if it is possible to place an up-down tile
    {
        place the up-down tile
        placeTiles(next x, next y)
        remove the tile
    }
    set the current square to not be covered
}
```

3.3 Combining the Programs

When combining the algorithms, the program generates the queens as usual, but the tiling is a bit different. Instead of tiling a regular chessboard, the program needs to tile a mutilated chessboard. In cases when the number of black and white squares are not equivalent, the chessboard cannot be tiled. To make the board mutilated, the squares on which the queens are placed need to be marked as covered.

Algorithm 3: Pseudo code for combined program.

```
placeTiles(current x, current y)
{
    if all tiles have been placed
    {
        add one to counter of solutions
        break
    }
    if the current square has been covered
    {
        placeTiles(next x, next y)
    }
    set the tile to be covered in the boolean array
    if it is possible to place a left-right tile
    {
        place the left-right tile
        placeTiles(next x, next y)
        remove the tile
    }
    if it is possible to place an up-down tile
    {
        place the up-down tile
```

```
        placeTiles(next x, next y)
        remove the tile
    }
    set the current square to not be covered
}
placeQueens(current column)
{
    if current column is n
    {
        if number of black squares = number of white squares
        {
            placeTiles(0, 0)
        }
        break
    }
    for all squares in the current column
    {
        check if the square is attacked by any queens in the same row
        check if the square is attacked by any queens on both diagonals
        if the square is not attacked
        {
            place the queen
            mark the square as covered
            placeQueens(next column)
            remove the queen
            mark the square as not covered
        }
    }
}
```

4 Results

For each $n \in [2, 15]$, we calculate the number of all solutions to the n-queens problem and the number of those that are tileable. From the results presented in Table 1, we conjectured that:

- $n \equiv 0, 1 \pmod 4$: Some but not all arrangements of queens can be tiled.
- $n \equiv 2, 3 \pmod 4$: None of the arrangements of queens can be tiled.

For a given n by n chessboard, the number of white squares is $\lceil \frac{n^2}{2} \rceil$, the number of black squares is $\lfloor \frac{n^2}{2} \rfloor$, and the number of all squares is n^2. A necessary condition to have a domino tiling is the number of white squares be equal to the number of black squares. If n is even, then the number of black and white queens will need to be $\frac{n}{2}$ each. If n is odd, then the number of white queens needs to be $\frac{n+1}{2}$ and the number of black queens needs to be $\frac{n-1}{2}$. Using these facts, we create Table 2 for different $n \pmod 4$.

Table 1: Exact number of solutions for placing n queens on n by n board.

n	All Solutions	Tileable Solutions
2	0	0
3	0	0
4	2	2
5	10	8
6	4	0
7	40	0
8	92	92
9	352	304
10	724	0
11	2,680	0
12	14,200	12,996
13	73,712	58,392
14	365,596	0
15	2,279,184	0

Table 2: Exact number of white/black queens and squares when placing n queens on n by n board to be tileable.

n	All Squares	White Squares	Black Squares	White Queens	Black Queens
$4k$	$16k^2$	$8k^2$	$8k^2$	$2k$	$2k$
$4k+1$	$16k^2+8k+1$	$8k^2+4k+1$	$8k^2+4k$	$2k+1$	$2k$
$4k+2$	$16k^2+16k+4$	$8k^2+8k+2$	$8k^2+8k+2$	$2k+1$	$2k+1$
$4k+3$	$16k^2+24k+9$	$8k^2+12k+5$	$8k^2+12k+4$	$2k+2$	$2k+1$

We then run the combined program to obtain all solutions and tiling arrangements when possible. The program checks if the conditions in Table 2 are satisfied before trying to tile the chessboard. For each, $n \in [2, 15]$, the program outputs the number of white/black queens along with the number of remaining white and black squares. It shows the number of solutions that satisfy these properties. The results for $n = 8$ are shown in Table 3, while results for $n = 15$ are displayed in Table 4.

Table 3: Possible numbers of white and black queens along with the possible numbers of remaining white and black squares and arrangements on an 8 by 8 chessboard.

White Queens	Black Queens	White Squares	Black Squares	Arrangements
4	4	28	28	92

Table 4: Possible numbers of white and black queens along with the possible numbers of remaining white and black squares and arrangements on an 15 by 15 chessboard.

White Queens	Black Queens	White Squares	Black Squares	Arrangements
5	10	108	102	33,912
7	8	106	104	1,739,936
9	6	104	106	504,776
11	4	102	108	560

5 Conclusion

In this paper, we studied the n-queens problem and the domino tiling problem numerically, created efficient programs to calculate these numbers, all for $n \in [2, 15]$. Based on these results, we have conjectured the following.

As long as the necessary condition of having equal white and black squares in a given n-queens solution is met, the remaining chessboard will be tileable. This is not always the case for other mutilated chessboards as they could have mutilations which split the board into groups of squares that are not tileable. In the example of n-queens, the queens, or mutilations, are spread out so far apart that the entire board is connected.

The number of black squares and number of white squares being equal sometimes occurs when $n \equiv 0, 1 \pmod 4$. On the other hand, for $n \equiv 2, 3 \pmod 4$, none of the solutions of n-queens satisfy this property of equivalence. Further research can be done to prove this.

6 Code and Data

The source code used to solve the two problems and obtain the results in the paper can be found at https://github.com/cks524/QED2022.

The tabulated results can be found in an excel sheet with tabs that has the data output for $n \in [2, 15]$. For each n, this sheet shows the number of solutions to n-queens and the number of solutions which are tileable. In addition, for every possible number of white and black queens, the data shows the white/black squares distribution and the number of solutions to n-queens that satisfy this property. For $n \leq 11$, there are tables that show the number of tilings of a solution and the frequency of that number.

References

[Bal60] W. W. Rouse Ball. The eight queens problem. *Mathematical Recreations and Essays*, pages 165–171, 1960.

[EST92] Cengiz Erbas, Seyed Sarkeshik, and Murat M. Tanik. Different perspectives of the n-queens problem. In *Proceedings of the 1992 ACM Annual Conference on Communications*, CSC '92, page 99–108, New York, NY, USA, 1992. Association for Computing Machinery.

[Kas61] P.W. Kasteleyn. The statistics of dimers on a lattice: I. the number of dimer arrangements on a quadratic lattice. *Physica*, 27(12):1209–1225, 1961.

[Sim21] Michael Simkin. The number of n-queens configurations, 2021.

[TF61] H. N. V. Temperley and Michael E. Fisher. Dimer problem in statistical mechanics-an exact result. *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics*, 6(68):1061–1063, 1961.

[Top82] Rodney W. Topor. Fundamental solutions of the eight queens problem. *BIT Numerical Mathematics*, 22:42–52, 1982.

[Wik22] Wikipedia. Domino tiling — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Domino%20tiling&oldid=1119365150`, 2022. [Online; accessed 22-November-2022].