A Novel Double Backtracking Approach to the N-Queens Problem in Three Dimensions

Article *in* International Journal of Computer Applications · July 2017

DOI: 10.5120/ijca2017914749

4 authors, including:

Pallavi Venkatesh
Sir M. Visvesvaraya Insititute of Technology
6 PUBLICATIONS **1** CITATION

International Journal of Computer Applications (0975 - 8887)
Volume 169 - No.5, July 2017

A Novel Double Backtracking Approach to the N-Queens Problem in Three Dimensions

Abhijith Chakiat
Department of Computer Science,
Sir M Visvesvaraya
Institute of Technology,
Bengaluru, Karnataka, India

Akhil Sudhakaran
Department of Computer Science,
Sir M Visvesvaraya
Institute of Technology,
Bengaluru, Karnataka, India

Abhinav A. Nair
Department of Computer Science,
Sir M Visvesvaraya
Institute of Technology,
Bengaluru, Karnataka, India

Pallavi Venkatesh
Department of Computer Science,
Faculty of Computer Science,
Sir M Visvesvaraya Institute of Technology,
Bengaluru, Karnataka, India

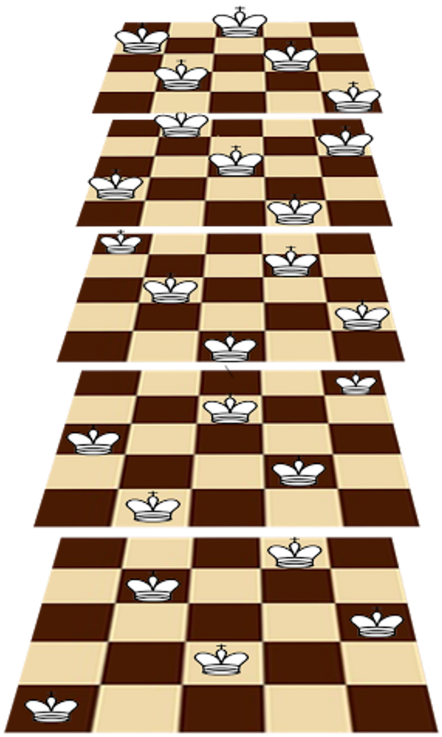

Fig. 1. Three dimensional N-Queens representation with 25 queens.

ABSTRACT

A well-known classic chessboard problem is that of placing N Queens on an N $\times$ N chessboard such that no two queens are able to attack each other (A Queen can attack in any direction, either in the same row, column or even diagonal). This is further generalized to three dimensions (N $\times$ N $\times$ N cube). An implementation of double backtracking algorithm to generate solutions for a certain value of N in a three dimensional space is discussed in this paper. The approach finds all the two dimensional solutions for a given N and records these. These two dimensional solutions are then stacked on top of one another such that no queen is present vertically on the same line as another queen. This builds up the three dimensional solution from the entire two dimensional solution set, picking one by one, using the general back tracking algorithm, with the constraint that no queen can attack another queen in the same plane, i.e same row, column or even diagonal, and no queen can attack another queen vertically on the same line across the planes of the cube constructed.

Keywords

N-Queens, Three-Dimensional N-Queens, Backtracking

1. INTRODUCTION

In 1848, Max Bezzel [1] published the eight queens puzzle. Since then this puzzle has caught the eye of many mathematicians. By 1850, Franz Nauck[2] published the first solution. He also generalized the puzzle to the N-Queens problem, with N queens on an N $\times$ N chessboard. The problem was to determine the placement of N queens on an N $\times$ N chess board such that no two queens can attack each other. A queen can attack another queen in the same column, row, or diagonal. Since then, many mathematicians, including Carl Friedrich Gauss[3], have worked on both the eight queens puzzle and its generalized N-Queens version. In 1874, S. Gunther[2] proposed a method using determinants to find solutions.

J.W.L. Glaisher[4] refined Gunther's approach by simplifying it. There has been different ways to represent the N-queens problem in higher dimensions, one such representation was depicted by S.P. Nudelman[5]. This paper focuses on the three dimensional representation of this classic problem.
The proposed definition of the three-dimensional N-queens problem follows the same definition of the two dimensional representation, where in, the two dimensional solutions stack up on top of one another to result in a N $\times$ N $\times$ N cube, and given two queens, they can attack one another if they lie on a common vertical hyperplane.
Fig 1. shows the representation of the problem in three dimensions, where in, 5 queens are placed in the 5 levels of the cube and, hence, produces 5 $\times$ 5 queens.

2. RELATED WORK

According to the arithmetic definition of the two dimensional N-queens problem given by Erbas *et al.*[6]: A set *A* represents a solution to the problem if for the Cartesian product of A, A $\times$ A, each element$<< i_k, j_k >, < i_l, j_l >>$ either satisfies all of the following conditions or fails all of them, for all l, k = 0 to N-1:

(1) $i_k \neq i_l$ (not on same column)
(2) $j_k \neq j_l$ (not on same row)
(3) $i_k + j_k \neq i_l + j_l$ (not on same diagonal)
(4) $i_k - j_k \neq i_l - j_l$ (not on same diagonal)

The implementation considers two queens in an attacking position if two are either in the same row, column or diagonal of the two dimensional plane they are in, or when they are vertically on top of each other in different two dimensional planes, i.e.: $i_h \neq j_h$ (not on the same height of the cube) where h is the vertical height of the cube. This approach guarantees N^2 Queens in the N $\times$ N $\times$ N cube.
As mentioned by Nudelman[5], in the original N queens problem, two queens located at (x_1,x_2) and (y_1,y_2) respectively, attack each other if one of the following conditions is satisfied.

(1) $x_1 \equiv y_1 (mod n)$
(2) $x_2 \equiv y_2 (mod n)$
(3) $x_1 + x_2 \equiv y_1 + y_2 (mod n)$
(4) $x_1 - x_2 \equiv y_1 - y_2 (mod n)$

In other words, the two queens attack each other if they are collinear on one of the four standard lines. The same paper provides a preposition that a d-dimensional queen attacks in $\frac{1}{2}(3^d$ - 1) hyperplanes. Nudelman[5] concludes by saying that if a complete solution exists for the d dimensional modular n chessboard problem, no prime less than 2d can divide n, but only when no prime less than 2^d divides n, the solution can be generated.
Barr[7] described the queens' attack lines in Cartesian d-space. In three dimensions, the following attack lines for a queen located at (i, j, k): The 6 two dimensional diagonals, and three axes:

(1) x - i = y - j
(2) x - i = j - y
(3) x - i = z - k
(4) x - i = k - z
(5) y - j = z - k
(6) y - j = k - z

x = i, y = j and z = k are the axes.
The 4 three dimensional diagonals are:

(1) x - i = y - j = z - k
(2) x - i = y - j = k - z
(3) x - i = j - y = z - k
(4) i - x = y - j = z - k

It is seen that the number of solutions for the traditional two-dimensional N-Queens problem can grow very fast, as shown in Fig. 2.

Number of Queens	Number of Solutions
3	0
4	2
5	10
6	4
7	40
8	92
9	352
10	724

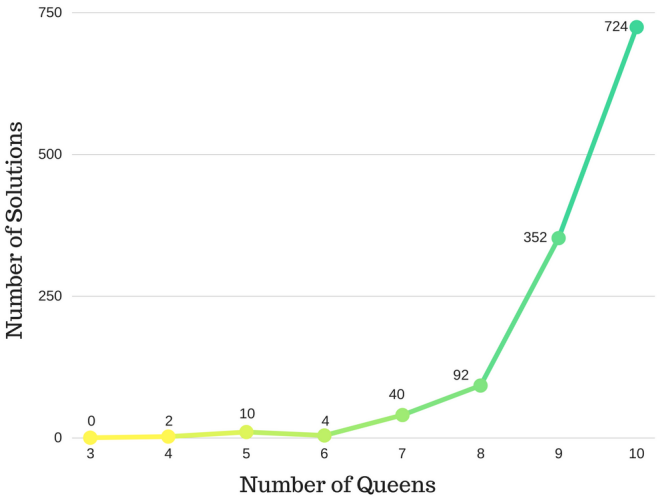

Fig. 2. Two dimensional N-Queens solutions graphically

3. PROPOSED MODEL

Multiple solutions (three dimensional) can be obtained for given value of N (if they exist), but the paper focuses only on obtaining a single solution to the three dimensional N-Queens problem. The major strategy is the Backtracking algorithm, which is used to obtain all the two dimensional solutions for a given value of N. The three dimensional solution is obtained by using backtracking again to stack up the two dimensional solution on top one another till the desired result is found. An abstract block representation of the proposed model is shown in Fig.3.

Since the paper focuses on the three dimensional solution of the N-Queens problem, the construction of the two dimensional solution is assumed and not discussed in detail.
For a given value of N, all the solutions to the classic N-Queens problem (two dimensional) is constructed and maintained, in what

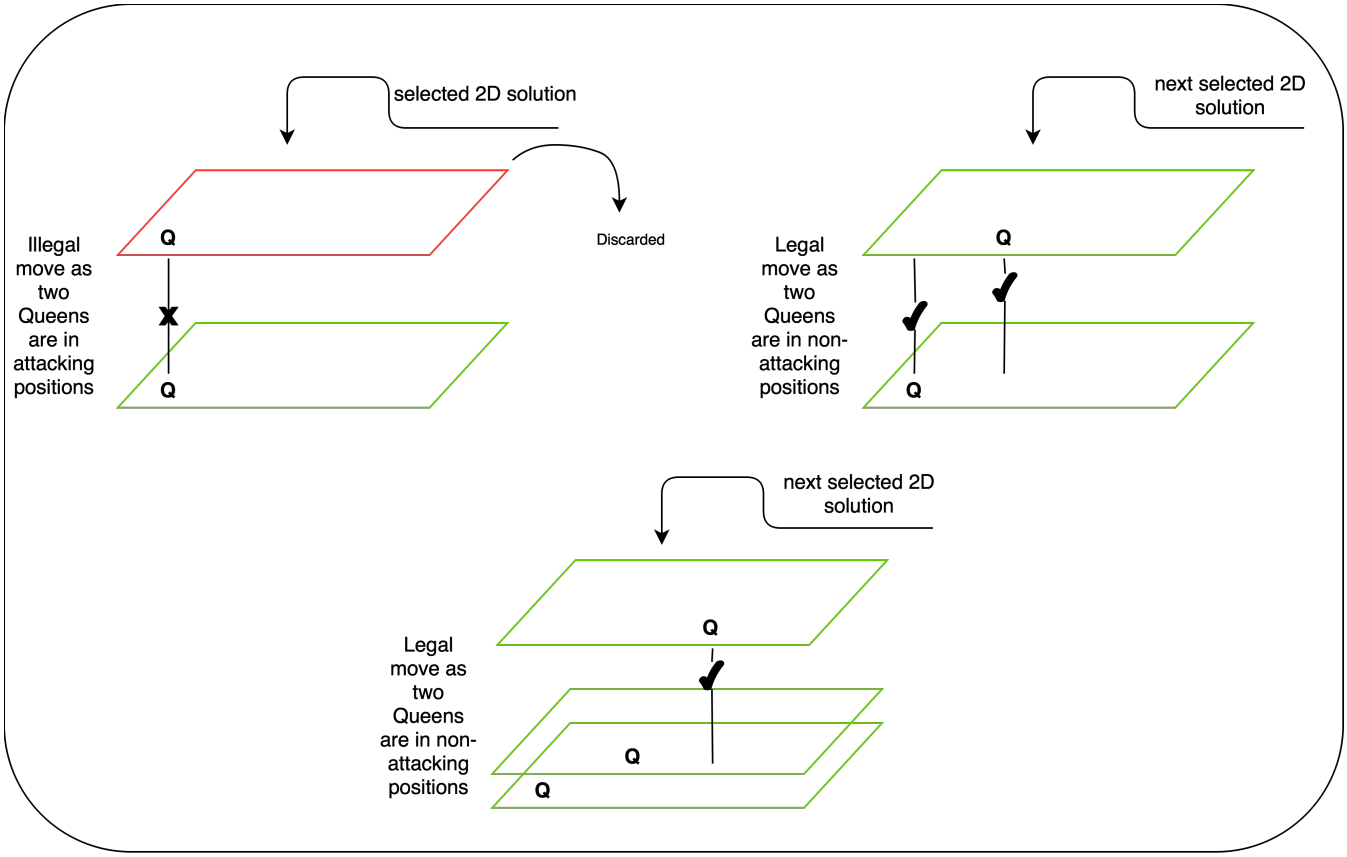

Fig. 4. Model of the three dimensional solution constructor

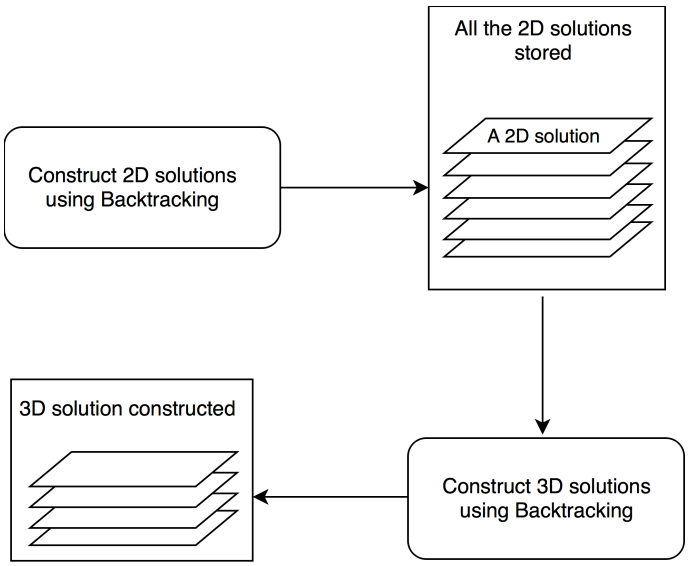

Fig. 3. Abstract model

the authors call a 2D solution stack.

The three dimensional solution constructor picks up a 2D dimensional solution from the stack containing all the solutions and starts building the three dimensional solution. It is done by placing one 2D solution over another and checking whether the move is legal according to the proposed representation of the problem, i.e whether a queen lies vertically on another queen across the previously selected 2D solutions. Of course, the first 2D solution is selected without any constraint. If the move is legal, then the selected 2D solution is placed on the previously constructed partial solution and a new solution is picked up from the stack of all 2D solutions.
If a particular solution is deemed illegal, it is discarded. If no solution is being found from the previously constructed partial solution, then the algorithm backtracks and replaces a previous selection with the next candidate available at that particular instant. The proposed approach stops when a single solution is constructed. It can be modified to continue till all such solutions are obtained.
Fig. 3 depicts how the backtracking algorithm selects the 2D solutions and constructs the final solution.

4. IMPLEMENTATION

The general Backtracking algorithm is the backbone of this approach to solve this problem. It is employed twice to construct

the desired solution.
The implementation model can be represented as shown in Fig.4, where a 2D solution is selected and is used to construct the 3D solution.
An abstract algorithm to find the solution of this problem can be stated as:

Algorithm 1 Algorithm for NQP in 3D

Input: N, size of grid in a single axis
Output: 3D solution
1: Backtrack and construct all 2D solutions for N and store in *allSolutions*
2: From *allSolutions* select the legal solutions and construct the *finalSolution* using Backtracking method
3: **return** $finalSolution$

allSolutions(Algorithm 1, line 2) contains all the possible two dimensional solutions for the given *N*.

Algorithm 2 3DNQP

Input: finalSolution, height
Output: boolean
Initialisation: t = *top*, of allSolutions
1: **if** $(height \geq N)$ **then**
2: return *true*
3: **end if**
LOOP Process
4: **for** $t = top$ to $size\_of\_allSolutions$ **do**
5: **if** $(isLegal(finalSolution, height, t)$ **then**
6: **for** $i = 1$ to N **do**
7: **for** $j = 1$ to N **do**
8: finalSolution[i][j][height] = allSolutions[i][j][t]
9: **end for**
10: **end for**
11: top = t + 1
12: **if** $(3DNQP(finalSolution, height + 1))$ **then**
13: return *true*
14: **end if**
15: **end if**
16: **end for**
17: **return** $false$

Algorithm 2 selects, one-by-one, the 2D solution from the top of the *allSolutions*(Algorithm 1, line 2) stack. It is then added to the *finalSolution*(Algorithm 1, line 2) stack if it is found legal and safe to be placed on the previously constructed solution (whether it can be safely placed above the previously selected 2D solutions in the finalSolution stack). If it is found unsafe or illegal, then it is discarded and the next available option is selected from the *allSolutions*(Algorithm 1, line 2) stack and the process is repeated. The legality of placing a 2D solution on previously selected solutions is governed by *Algorithm 3*.

5. RESULT

The example taken in the introduction section can be analyzed, where the problem was to find N × N queens' positions in an N×N×N grid.
The paper proposes a two dimensional representation (Fig. 5) of this solution as it is much easier to comprehend, than a three dimensional one.

Algorithm 3 isLegal

Input: finalSolution, height, t
Output: boolean
1: **if** $(height == N)$ **then**
2: return *true*
3: **else**
4: *LOOP Process*
5: **for** $i = 1$ to N **do**
6: **for** $j = 1$ to N **do**
7: **if** $(allSolutions[i][j][t] == true)$ **then**
8: **for** $h = height - 1$ to 1 **do**
9: **if** $(finalSolution[i][j][h])$ **then**
10: return *false*
11: **end if**
12: **end for**
13: **end if**
14: **end for**
15: **end for**
16: **end if**
17: **return** $false$

1	2	3	4	5
4	5	1	2	3
2	3	4	5	1
5	1	2	3	4
3	4	5	1	2

Fig. 5. Solution to the three dimensional N Queen problem with 25 queens. The coloured numbers indicate the level at which the queen is placed

Each digit represents a queen placed in the N×N×N grid. The number corresponds to the level in which the queen is placed. Level *1* corresponds to the lowest level or the bottom face of the cube formed and level *5* corresponds to the top-most level or the top face of the cube. It is seen that no queen is in an attacking position with other queens in the same level. This agrees with the two dimensional solution of N-Queens with *N = 5*. No queen also attacks any other queen that is in the same vertical hyperplane. This goes with the representation of the problem as stated earlier. Fig. 1. from the introduction section depicts the same solution in a three - dimensional fashion.
The numbers are also coloured for ease of understanding. The same coloured numbers belong to the same level.
It is also understood that, the 3D solution might exist if and only if, for a given N, the number of two dimensional solutions is greater than or equal to N. Consider N = 4, for this value of N, there are

only two 2D solutions hence a 3D solution is impossible.
The general backtracking algorithm for the N-queens problem has a time complexity of O(N!). Back tracking is used twice in the proposed approach, hence an upper bound of O(N!) is seen.

6. CONCLUSION

The paper states how the proposed approach to the three dimensional N-Queens problem works. The backtracking method is used twice to get the desired result, where in, once it is used to find the various two dimensional solutions for the given value of N, and another to construct a single three dimensional solution by stacking one two dimensional solution on another subject to the constraints mentioned. Using this method it is possible to effectively retrace back to the previous solution if the present two dimensional solution does not fit into the solution, hence starting at the point where the solution was apt and not going to the beginning to start the process all over again. The proposed algorithm stops comparing once the required solution from the set of 2-D N queens solutions is found.

7. FUTURE WORK

This paper discusses the implementation of the three dimensional N-Queens problem without any significant optimization strategies. One optimization strategy that can be used while computing all the two dimensional solutions would be to use the Ant Colony Optimisation (ACO)[8], which was first developed by Dorigo[9]. The solution proposed in [8] works efficiently for the 8-queen problem and it can be easily extended to large values of n because of the simplistic model of the search space.
[10] presents a new approach for solving this problem. A quantum-inspired differential evolution algorithm, which is a hybridization of two well known algorithms, DEA and QGA, is discussed in [10].
The discussed method constructs only a single solution for the three dimensional problem. It can further be built to produce all the solutions, which might be computationally heavy, hence it should be coupled with some optimization strategies mentioned above.

8. REFERENCES

[1] F.W.M. Bezzel. Proposal of eight queens problem. *Berliner Schachzeitung*, 1848.

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

[3] C.F.Gauss und H.C.Schumacher. 6, 1865. CF editor.

[4] J W L. the problem of the eight queens. *Philosophical Magazine*, (4):457–467, November 1874.

[5] S.P.Nudelman. The modular n-queens problem in higher dimensions. *Discrete Math.*, 146:159–167, 1995.

[6] C.Erbas and M.M. Tanik S. Sarkeshik. Different perspectives of the n-queens problem. *Proceedings of the 1992 ACM Annual Conference on Communications, ACM Press*, page 99108, 1992.

[7] J. Barr and S. Rao. Some observations about the n-queens problem in higher dimensions. *Midwest Instructions and Computing Symposium*, 2004.

[8] M. Sharif S. Khan, M. Bilal, M.Sajid, and R.Baig. Solution of n-queen problem using aco. *International Multitopic Conference, IEEE*, 2009.

[9] M.Dorigo. Optimization, learning and natural algorithms. In *PhD thesis*, 1992.

[10] H.Talbi A.Draa, S.Meshoul and Batouche. *A quantum-inspired differential evolution algorithm for solving the N-queens problem*, page 2127. 2010.