ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

REVIEW ON N-QUEEN OPTIMIZATION USING TUNED HYBRID TECHNIQUE

Er. Vishal Khanna*, Er. Sarvesh Chopra
* Research Scholar CT group of Institutions, Jalandhar, Punjab, India
Asst. Prof. CT group of Institutions, Jalandhar, Punjab, India

DOI:

ABSTRACT

Provide an effective approach to improve the well-known and most used technique: backtracking, on N-queen problem which is to place the 'n' numbers of queens on a chess board so that neighbor queens cannot contradict each other vertically, Horizontally and diagonally if contradiction occurs the no of trails will goes on and to increase the performance by removing the threatening cells in order to decrease the number of trails and number of error steps.

KEYWORDS: Queens, N-queens, 8 Queen, Backtracking, Backtracking and Set, Sets, Optimization, Hybrid Technique.

INTRODUCTION

N-Queens dates back to the 19th century (studied by Gauss) Classical combinatorial problem, widely used as a benchmark because of its simple and regular structure Problem involves placing N queens on an N × N chessboard such that no queen can attack any other Benchmark code versions include finding the first solution and finding all solutions. Input for this System: A Positive integer n. Task for this System: Place n queens on an n by n chessboard so that no two queens attack each other (on same row, column, diagonal), or report that this is impossible. Solving particular problem for the different values of n=1, 2, 3, 4…n. [1]

Process of n-queen:

1. Suppose you have 8 chess queens and a chess board of Size 8*8.
2. Can the queens be placed on the board so that no two queens are attacking each other?

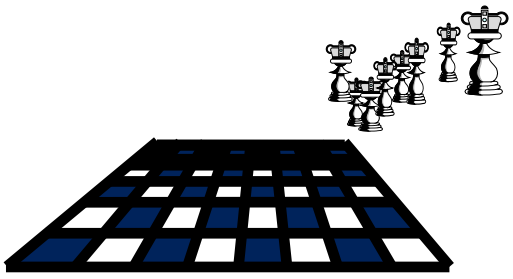

Fig: 1.1 Chess Board of Size 8*8 with 8 Queens. [2]

3. Two queens are not allowed in the same row.
4. Two queens are not allowed in the same row, or in the same column.
5. Two queens are not allowed in the same row, or in the same column, or along the same diagonal.
6. The number of queens and the size of the board can vary.
7. It seems hard to generate one valid placement.
8. But it is easy to check whether a placement is valid or not.
9. We will write a program which tries to find a way to place N queens on an N x N chess board.
10. The program uses a stack to keep track of where each queen is placed.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

11. Each time the program decides to place a queen on the board, the position of the new queen is stored in a record which is placed in the stack.
12. We also have an integer variable to keep track of how many rows have been filled so far.
13. Each time we try to place a new queen in the next row, we start by placing the queen in the first column.
14. If there is a conflict with another queen, then we shift the new queen to the next column.
15. If another conflict occurs, the queen is shifted rightward again.
16. When there are no conflicts, we stop and add one to the value of filled.
17. Let's look at the third row. The first position we try has a conflict.
18. So we shift to column 2. But another conflict arises.
19. Then we shift to the third column. Yet another conflict arises.
20. We shift to column 4. There's still a conflict in column 4, so we try to shift rightward again.
21. When we run out of room in a row: pop the stack, reduce filled by 1 and continue working on the previous row.
22. Now we continue working on row 2, shifting the queen to the right.
23. This position has no conflicts, so we can increase filled by 1, and move to row 3.
24. In row 3, we start again at the first column.

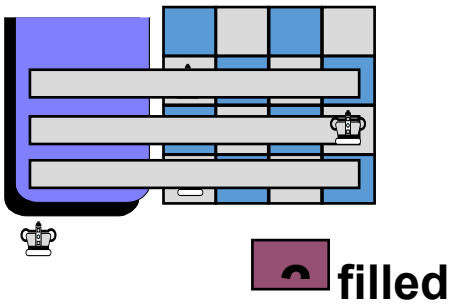

Fig: 1.2 Move queens in next row and repeat process. [2]

APPLICATIONS OF N-QUEEN

There are Variety of N-Queen Applications that is deal with the daily life and real world problems. Some of these are given below:

1. VLSI Testing.
2. Traffic control.
3. Deadlock Prevention.
4. Image Processing.
5. Motion Estimation.
6. Register Allocation.[6]

Optimization:

Optimization is the process of identifying the best solution among a set of alternatives. Single objective optimization employs a single criterion for identifying the best solution among a set of alternatives.

Mathematical Equations:

These brute-force algorithms to count the number of solutions are computationally manageable for n = 8, but would be intractable for problems of $n \geq 20$, as $20! = 2.433 \times 10^{18}$. If the goal is to find a single solution then explicit solutions exist for all $n \geq 4$, requiring no combinatorial search whatsoever.[4] The explicit solutions exhibit stair-stepped patterns, as in the following examples for $n = 8$, 9 and 10.

The examples above can be obtained with the following formulas. Let (i, j) be the square in column i and row j on the $n \times n$ chessboard, k an integer.

1. If n is even and $n \neq 6k + 2$, then place queens at $(i, 2i)$ and $(n/2 + i, 2i - 1)$ for $i = 1, 2, ..., n / 2$.
2. If n is even and $n \neq 6k$, then place queens at $(i, 1 + (2i + n/2 - 3 \pmod n))$ and $(n + 1 - i, n - (2i + n/2 - 3 \pmod n))$ for $i = 1, 2, \ldots n / 2$.
3. If n is odd, then use one of the patterns above for $(n - 1)$ and add a queen at (n, n).[2]

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

Backtracking and Set:

Backtracking is a technique used to solve problems with a large search space, by systematically trying and eliminating possibilities. Backtracking is a methodical way of trying out various sequences of decisions, until you find one that "works". When we carry out backtracking, an easy way to visualize what is going on is a tree that shows all the different possibilities that have been tried. On the board we will show a visual representation of solving the 4 Queens problem (placing 4 queens on a 4x4 board where no two attack one another). The neat thing about coding up backtracking is that it can be done recursively, without having to do all the bookkeeping at once. Instead, the stack or recursive calls does most of the book keeping track of which queens we've placed, and which combinations we've tried so far, etc. sets are used as input to enhance the
Backtracking technique by removing threatening cells. [1]

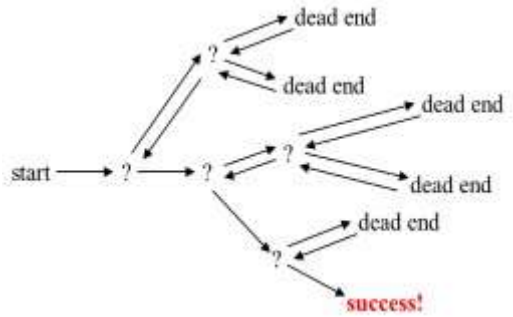

Fig: 1.3 Process of Backtracking to find success. [1]

Proposed Algorithm:

In this paper two algorithms are discussed these are given below:

1) Backtracking
2) Backtracking and Sets

Backtracking: Backtracking is a form of recursion. The usual scenario is that you are faced with a number of options, and you must choose one of these. After you make your choice you will get a new set of options; just what set of options you get depends on what choice you made. This procedure is repeated over and over until you reach a final state. If you made a good sequence of choices, your final state is a *goal state;* if you didn't, it isn't. Conceptually, you start at the root of a tree; the tree probably has some good leaves and some bad leaves, though it may be that the leaves are all good or all bad. You want to get to a good leaf. At each node, beginning with the root, you choose one of its children to move to, and you keep this up until you get to a leaf.

Suppose you get to a bad leaf. You can *backtrack* to continue the search for a good leaf by revoking your *most recent* choice, and trying out the next option in that set of options. If you run out of options, revoke the choice that got you here, and try another choice at that node. If you end up at the root with no options left, there are no good leaves to be found.[1]
This needs an example.

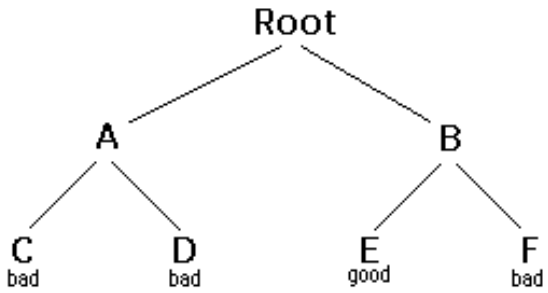

Fig: 1.4 Backtracking recursive processes. [1]

Pseudo Code:

```
Boolean solve (Node n)
{
        if n is a leaf node
```

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

```
{
        if the leaf is a goal node, return true
else return false
}
else
{
for each child c of n
{
if solve(c) succeeds, return true
}
Return false
}
}
```

Algorithm of Backtracking:

1. Place the queens column wise, start from the left most column
2. If all queens are placed.
    1. Return true and print the solution matrix.
3. Else
    1. Try all the rows in the current column.
    2. Check if queen can be placed here safely if yes mark the current cell in solution matrix as 1 and try to solve the rest of the problem recursively.
    3. If placing the queen in above step leads to the solution return true.
    4. If placing the queen in above step does not lead to the solution, BACKTRACK, mark the current cell in solution matrix as 0 and return false.
4. If all the rows are tried and nothing worked, return false and print NO SOLUTION.

Backtracking and Sets Algorithm: The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, following 0is a solution for 4 Queen Problem. The expected output is a binary matrix which has 1s for the blocks where queens are placed. For example following is the output matrix for above 4 queen solution.

{0, 1, 0, 0}
{0, 0, 0, 1}
{1, 0, 0, 0}
{0, 0, 1, 0}

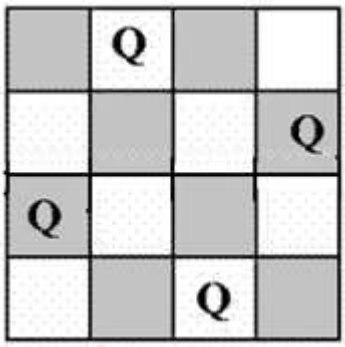

Fig: 1.5 Backtrack and sets output. [1]

Algorithm of Backtracking and Sets:

1. Place the queens column wise, start from the left most column
2. If all queens are placed.
    1. Return true and print the solution matrix.
3. Else
    1. Try all the rows in the current column.
    2. Check if queen can be placed here safely if yes mark the current cell in solution matrix as 1, provide set input and try to solve the rest of the problem recursively.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

3. If placing the queen in above step leads to the solution return true.
4. If placing the queen in above step does not lead to the solution, BACKTRACK, mark the current cell in solution matrix as 0 and return false.

If all the rows are tried and nothing worked, return false and print NO SOLUTION. [1]

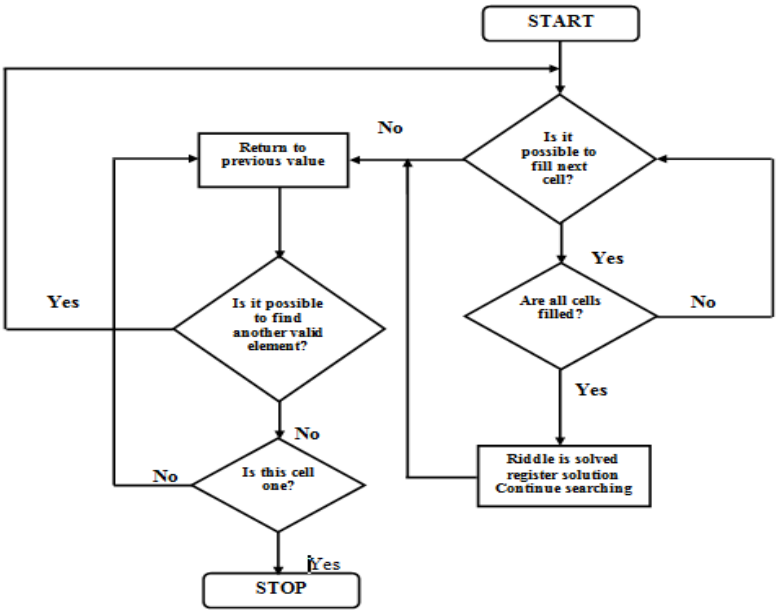

Fig: 1.6 Methodology of solving n-queens problem by Backtracking. [1]

RESULTS

Time complexity is measured with no of queens and results of backtracking and backtracking and sets in nanoseconds.

Table: 1 Time Complexity based on the number of Queens (n). [1]

S.No.	No of Queens (n)	Backtracking Algorithm(ns)	Backtracking and Sets Algorithm(ns)
1	4	4570122564	3460659861
2	5	4734813274	3572926461
3	6	5054721111	3586741016
4	7	5153144101	3596822619
5	8	5183487472	3612683286
6	9	5212678805	3619734391
7	10	6142798810	3879151333

Space complexity is measured with no of queens and results of backtracking and backtracking and sets in nanoseconds.

Table: 2 Space Complexity based on the number of Queens(n). [1]

S.No.	No of Queens (n)	Backtracking Algorithm(mb)	Backtracking and Sets Algorithm(mb)
1	4	2	1
2	5	2	1
3	6	3	2
4	7	3	2
5	8	4	2
6	9	4	3
7	10	5	3

System Configuration: Intel Celeron Processor N-3050, Intel HD Graphics, 4 GB RAM, 500 GB Hard Disk.

CONCLUSION AND FUTURE SCOPE

In this paper, we demonstrate that the efficiency of the traditional backtracking algorithm may be improved by the use of a hybrid approach taking advantage of sets to reduce the number of trials and error attempts. Time taken to solve the n-queen problem in the backtracking is more than that of the Tuned hybrid technique. Space taken to solve the n-queen problem in the backtracking is more than that of the Tuned hybrid technique. Complexity Analysis can be improved using different algorithms and that approach will be applied on the one of the applications of the N-Queen Problem to obtain the fast and better solution. Complexity Analysis can be based in the time, space, convergence-rate and conflict-minimization.

REFERENCES

[1] Sarkan Guldal and Veronica Baugh, "N-Queens Solving Algorithm by Sets and Backtracking", IEEE Southeast Conference, pp.125-129, 2016.
[2] B Documentaries, "Full Solution of N-Queen Problem O Reilly", http://oreillynqueenproblem.blogspot.in, 3-Sept-2016.
[3] Amarbir Singh and Sandeep Singh Dhillon, "A Comparative Study of Algorithms for N-Queen Problem", International Journal of Advance Foundation and Research in Science and Engineering , Vol.1, Special Issue, pp.1-4, 2015.
[4] Soham Mukherjee, Santanu Datta, Pramit Brata Chanda and Pratik Pathak, "Comparative Study of Different Algorithms To Solve N-Queens Problem", International Journal of Foundations of Computer Science and Technology, Vol.5, Issue.2, pp.15-27, 2015.
[5] Ahmed S. Farhan , Wadhan Z. Tareq and Fouad H. Awad, "Solving N-Queen Problem using Genetic Algorithm ", International Journal of Computer Applications, Vol.122, Issue.12, pp.11-14, 2015.
[6] Vikas Thada and Shivali Dhaka, "Performance Analysis of N-Queen Problem using Backtracking Algorithm Techniques ", International Journal of Computer Applications, Vol.102, Issue.7, pp. 26-29, 2014.
[7] Ellips Masehian, Hossein Akbaripour and Nasrin Mohabbati-Kalejahi,"Solving the n-Queens Problem Using a Tuned Hybrid Imperialist Competitive Algorithm ", The International Arab Journal of Information Technology, Vol.11, Issue.6, pp.550-559, 2014.
[8] Vishal Kesri and Manoj Kumar Mishra, "A new approach to solve n-queens problem based on series", International Journal of Engineering, Research and Applications, Vol.3, Issue.3, pp.1349-1349, 2013.
[9] Ram Gopal Sharma and Bright Keswani, "Implementation of N-Queens Puzzle using Meta-Heuristic Algorithm (Cuckoo Search)" International Journal of Latest Trends in Engineering and Technology, Vol. 2, Issue. 3, pp. 343-347, 2013.
[10] S.Pothumani, "Solving N-Queen Problem using Various Algorithms-A Survey", International Journal of Advance Research in Computer Science and Software Engineering, Vol. 3, Issue. 2, pp. 247-250, 2013.
[11] Farhad Soleimanian, Bahareh Seyyedi and Golriz Feyziour, "A New Solution for N-Queens Problem using Blind Approaches: DFS and BFS Algorithms", International Journal of Computer Applications, Vol.53, Issue.1, pp.45-48, 2012.
[12] Vishal Kesri, Vaibhav Kesri and Prasant Ku. Pattnaik, "A Unique Solution for N-Queen Problem", International Journal of Computer Applications, Vol.43, Issue.12, pp.13-19, 2012.
[13] Baolei Gu," Research and Realization of N-Queens Problem Based on the Logic Language Prolog", Springer Computational Intelligence and Intelligent System, Vol.4, Issue.1, pp. 50-56, 2012.
[14] Aftab Ahmed, Attique Shah, Kamran Ali Sani and Abdul Hussain Shah Bukhari," International Journal of Advance Computer Science and Technology" Vol.1, Issue.2, pp. 57-63, 2012.
[15] Jun Zhang and Zili Zhang, "An Algebraic method for the n-Queen Problems based on Permutation Operation Group", International Journal of Computer Networks and Information Security, Vol.3, pp.19-25, 2011.
[16] Jordan Bell and Brett Stevens, "A Survey of Known results and research areas for n-queens" Discrete Mathematics Science Direct, Vol.309, Issue.1, pp.1-31, 2008.
[17] H. Ahrabian, A. Mirzaei and A.Nowzari-Dalini," A DNA Sticker Algorithm for Solving N-Queen Problem", International Journal of Computer Science and Applications, Vol.5, Issue.3, pp.12-22, 2006.
[18] Chung-Neng Wang, Shin-Wei Yang, Chi-Min Liu and Tihao Chiang, "IEEE Transactions on Circuits and System for Video Technology", Vol.14, Issue.4, pp.429-440, 2004.
[19] Marko Bozikovic, Marin Golub and Leo Budin, "Solving N-Queen Problem Using Global Parallel Genetic Algorithm", European Conference Ljubljana Slovenia, pp.11-17, 2003.
[20] Roc Sosic and Jun Gu, "Fast Search Algorithms for the N-Queen Problem", IEEE Transactions on Systems, Man, and Cybernetics, Vol.21, Issue.6, pp.1572-1576, 1991.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

[21] Rok Sosic and Jun Gu, "A Polynomial Time Algorithm for N-Queens Problem", Special Interest Group on Artificial Intelligence, vol.1, Issue.3, pp.7-14, 1990.