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

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

COMPARATIVE ANALYSIS OF BACKTRACKING, TUNED HYBRID TECHNIQUE AND GENETIC ALGORITHM FOR OPTIMIZATION OF N-QUEENS PROBLEM

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: 10.5281/zenodo.438095

ABSTRACT

Comparative analysis for N-Queens problem by using various techniques: backtracking, genetic algorithm and tuned hybrid technique. This NP hard problem states that “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 number of trails will go 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 Queens, Backtracking, Backtracking and Set, Sets, Optimization, Tuned Hybrid Technique, Genetic Algorithm, Chromosome, Mutation, Selection, Crossover, Recombination.

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 Queens 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-Queens:

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?

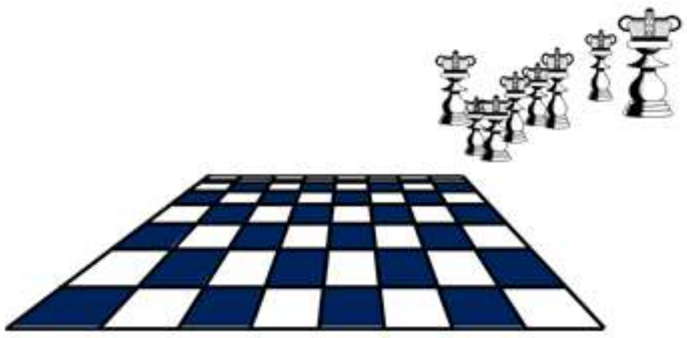


Fig: 1.1 Chess Board of Size 8*8 with 8 Queens. [1]

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 Queens is placed.
11. Each time the program decides to place a Queens on the board, the position of the new Queens 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 Queens in the next row, we start by placing the Queens in the first column.
14. If there is a conflict with another Queens, then we shift the new Queens to the next column.
15. If another conflict occurs, the Queens 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 Queens 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.[1]

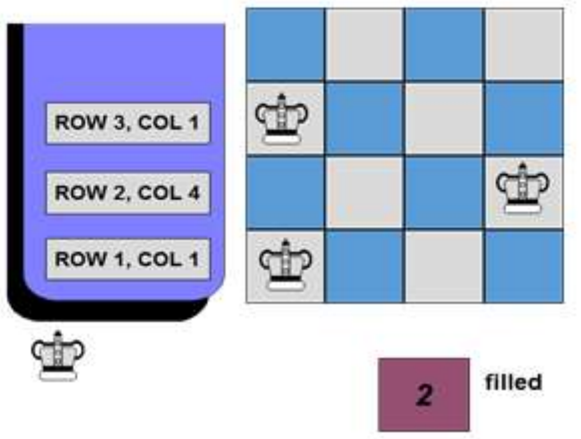


Fig: 1.2 Move Queens in next row and repeat process. [1]

APPLICATIONS OF N-QUEENS

There are Variety of N-Queens 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. [1]

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 $i = 1, 2, \dots n / 2$.
3. If n is odd, then use one of the patterns above for $(n - 1)$ and add a Queens at (n, n).[2]

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]

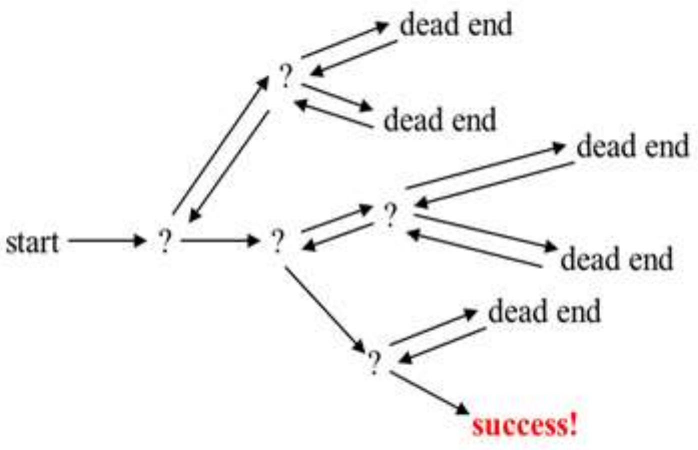


Fig: 1.3 Process of Backtracking to find success. [1]

Proposed Algorithm:

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

1) Backtracking
2) Backtracking and Sets
3) Genetic Algorithm

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.

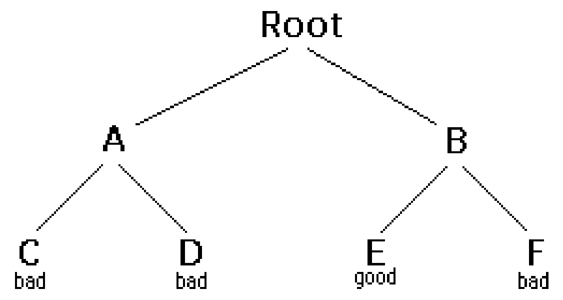


Fig: 1.4 Backtracking recursive processes. [1]

Pseudo Code:

```
Boolean solve (Node n)
{
        if n is a leaf node
        {
                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 Queens 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 Queens in above step leads to the solution return true.
    4. If placing the Queens 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 Queens 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 Queens 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 Queens Solution.

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

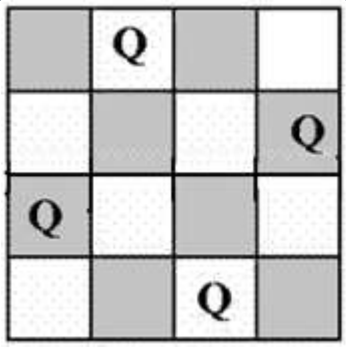


Fig: 1.5 Backtrack and sets output. [1]

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 Queens 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.
    3. If placing the Queens in above step leads to the solution return true.
    4. If placing the Queens 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]
```

Genetic Algorithm: Genetic algorithms (GA) are randomized search and optimization techniques that are motivated by the principals of natural selection and evolution processes. The GA was introduced by John Holland in 1975. He was inspired by the concept of Darwinian's principle of survival of the fittest individuals and natural selection and developed the theory of genetic algorithm. Since then many researchers are using this algorithm to solve various optimization problems effectively. [3]

Steps of Genetic Algorithms

In GA, the roles of initialization and recombination operators are very well defined. The initialization operator identifies the direction of search and recombination operator generates new regions for search. GA first starts a generation with an initial population. The initial population is generated with a number of chromosomes. The chromosomes are made up with a number of genes. For clustering problem, a chromosome is considered to be a clustering solution, and a gene of a chromosome is considered to be the center of a cluster. After initialization, in order to selection, an objective/fitness function is applied on each chromosome that identifies the goodness of a chromosome. Biologically inspired operators: crossover and mutation are then applied to the population in order to solve the clustering problem. At the end of each generation, GA applies an elitist operation where the newly generated populations are compared with the previous population. All these inter-related parameters and operators influence the performance of a GA. The processes of selection, crossover, mutation and elitist operation are continued for a fixed number of generations or until a termination condition is satisfied.

Step 1: Population initialization: The first step of GA is to initialize population. To initialize population, size is one of the important things that need to be considered. The higher number of the population contributes towards a good clustering solution. Therefore, some GA-based clustering techniques use big population for the clustering solution. Traditionally, the numbers of genes of a chromosome are generated randomly in the initial population. The records also selected randomly to form genes. However, a careful selection of genes can create an initial population containing high-quality chromosomes. Having high-quality chromosomes in a population increases the possibility of getting better quality clustering solution at the end of genetic processing. Therefore, some GA-based clustering techniques generate high-quality chromosomes in the initial population.

Step-2: Selection: For the next genetic operator such as crossover and mutation GA selects chromosomes based on their fitness/objective function. There are various methods to calculate the fitness such as Davis-Bouldin (DB) Index, Sum of the Squared Error (SSE), Silhouette Coefficient and COSEC.

Step-3: Crossover: Crossover is an important step in GA where a pair of chromosomes swaps their segments/genes to each other and generates a pair of new offspring chromosomes. Typically, there are many selection criteria such as the roulette wheel, rank-based wheel and random selections are used to select the chromosome pair for a crossover operation. Some GA-based clustering techniques use roulette wheel selection where the best chromosome (which is available in the current population) is chosen as one chromosome of the pair. The second chromosome of the pair is selected using the roulette wheel technique. Blas et al. use the rank-based wheel selection. In the rank-based wheel selection, the chromosomes are first sorted based on their quality and then the pair of chromosomes are chosen based on the position of a chromosome in the rank. Once the pair of chromosomes is selected, GA then applies crossover operation on each pair of chromosomes. There are many approaches to performing crossover between a pair of chromosomes such as a single point, multi-point, arithmetic, path-based and heuristic. In the single-point crossover, each chromosome of a pair is divided into two parts at a random point between two genes. The left part (having one or more genes) of one chromosome of a pair joins the right part of other chromosomes (having one or more genes) and form an offspring chromosome. In the multi-point crossover, each chromosome of a pair is divided into multiple parts and then swaps their part to each other and generates new offspring chromosome. In the path-based crossover two parents, chromosomes create a path between them. Two points are then selected from the obtained path as offspring chromosomes. The heuristic crossover uses the fitness values of two parents where the worst parent slightly moves towards the best parent.

Step- 4: Mutation: Mutation randomly changes one or more genes of a selected chromosome with a probability equal to the mutation rate. There are many approaches for mutation such as such as division and absorption, insertion, deletion, perturbation and movement. The division operation divides one cluster of a chromosome into two clusters. The absorption operation merges two clusters of a chromosome into one cluster. The perturb mutate randomly selects a cluster.

Advantages and disadvantages of genetic algorithms
At the end of the discussion about genetic algorithm improvements, we will list some of the attractive advantages and also some disadvantages of genetic algorithms:

Advantages:
1. It can solve every optimization problem which can be escribed with the chromosome encoding.
2. It solves problems with multiple solutions.
3. Since the genetic algorithm execution technique is not dependent on the error surface, we can solve multi-dimensional, non-differential, non-continuous, and even non-parametrical problems.
4. Structural genetic algorithm gives us the possibility to solve the solution structure and solution parameter problems at the same time by means of genetic algorithm.
5. Genetic algorithm is a method which is very easy to understand and it practically does not demand the knowledge of mathematics.
6. Genetic algorithms are easily transferred to existing simulations and models.

Disadvantages:
1. Certain optimization problems (they are called variant problems) cannot be solved by means of genetic algorithms. This occurs due to poorly known fitness functions which generate bad chromosome blocks in spite of the fact that only good chromosome blocks cross-over.
2. There is no absolute assurance that a genetic algorithm will find a global optimum. It happens very often when the populations have a lot of subjects.
3. Like other artificial intelligence techniques, the genetic algorithm cannot assure constant optimization response times. Even more, the difference between the shortest and the longest optimization response time is much larger than with conventional gradient methods. This unfortunate genetic algorithm property limits the genetic algorithms' use in real time applications.
4. Genetic algorithm applications in controls which are performed in real time are limited because of random solutions and convergence, in other words this means that the entire population is improving, but this could not be said for an individual within this population. Therefore, it is unreasonable to use genetic algorithms for on-line controls in real systems without testing them first on a simulation model. [3]

Algorithm of Genetic Algorithm

1. Instance
2. Size of population
3. Rate of elitism
4. Rate of mutation
5. Number of iterations [3]

Output: Optimal solutions

METHODOLOGY

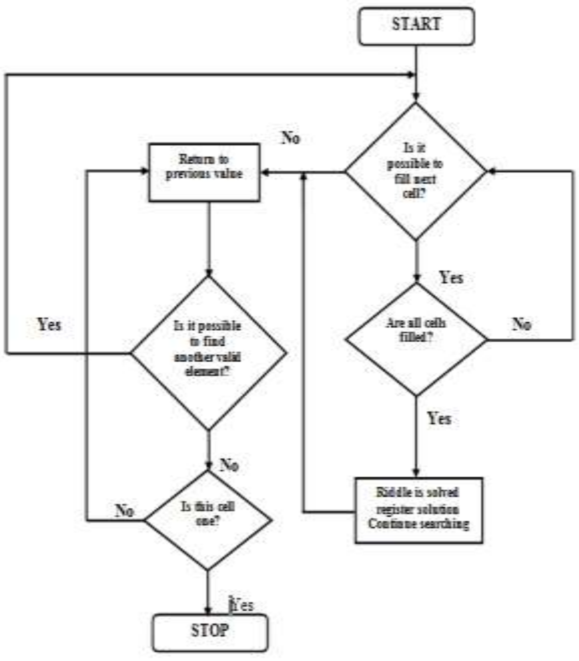
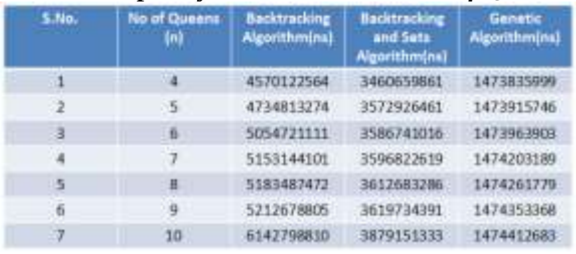

START
Is it possible to fill next cell?
No
Return to previous value
Yes
Are all cells filled?
No
Yes
Riddle is solved register solution Continue searching
Yes
Is it possible to find another valid element?
No
Is this cell one?
No
Yes
STOP

Fig: 1.6 Methodology of solving n-Queens problem. [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]

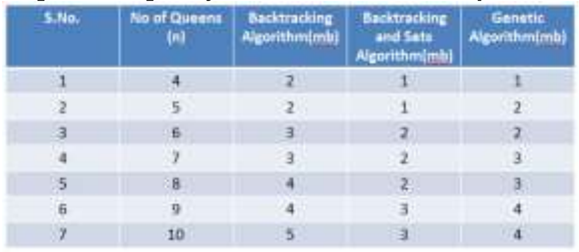


S.No.	No of Queens (n)	Backtracking Algorithm(ns)	Backtracking and Sets Algorithm(ns)	Genetic Algorithm(ns)
1	4	4570122564	3460659861	1473835999
2	5	4734813274	3572926461	1473915746
3	6	5054721111	3586741016	1473963903
4	7	5153144101	3596822619	1474203189
5	8	5183487472	3612683286	1474261779
6	9	5212678805	3619734391	1474353368
7	10	6142798810	3879151333	1474412683

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

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

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)	Genetic Algorithm(mb)
1	4	2	1	1
2	5	2	1	2
3	6	3	2	2
4	7	3	2	3
5	8	4	2	3
6	9	4	3	4
7	10	5	3	4

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 comparative analysis of three most used techniques for solving the n-Queens problem on the basis of parameters viz time and space. Time taken to solve the n-Queens problem in the backtracking and tuned hybrid technique is more than that of the genetic. Space taken to solve the n-Queens problem in the backtracking and tuned hybrid technique is more than that of the genetic. Complexity Analysis can be improved using different algorithms and that approach will be applied on the one of the applications of the N-Queens Problem to obtain the fast and better solution. Complexity Analysis can be based in the convergence-rate and conflict-minimization.

REFERENCES

[1] Vishal Khanna and Sarvesh Chopra, Review On N-Queens Optimization Using Tuned Hybrid Technique, International Journal Of Engineering Sciences & Research Technology, pp.62-68, Vol.6, Issue.3, 2017.

[2] Sarkan Guldal and Veronica Baugh, "N-Queens Solving -Algorithm by Sets and Backtracking", IEEE Southeast Conference, pp.125-129, 2016.

[3] A. H. Beg, Md Zahidul Islam, Advantages and Limitations of Genetic Algorithms for Clustering Records, 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA), pp.2478-2483, 2016

[4] Anuneet Kumar Dubey, Vijayan Ellappan, Rakesh Paul, Vishal Chopra, Comparative Analysis Of Backtracking And Genetic Algorithm In N Queens's Problem, International Journal of Pharmacy & Technology, pp.25618-25623, Vol.8, Issue.4,2016

[5] B Documentaries, "Full Solution of N-Queens Problem O Reilly", http://oreillynQueensproblem.blogspot.in, 3-Sept-2016.

[6] Amarbir Singh and Sandeep Singh Dhillon, "A Comparative Study of Algorithms for N-Queens Problem", International Journal of Advance Foundation and Research in Science and Engineering , Vol.1, Special Issue, pp.1-4, 2015.

[7] 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.

[8] Ahmed S. Farhan , Wadhan Z. Tareq and Fouad H. Awad, "Solving N-Queens Problem using Genetic Algorithm ", International Journal of Computer Applications, Vol.122, Issue.12, pp.11-14, 2015.

[9] Vikas Thada and Shivali Dhaka, "Performance Analysis of N-Queens Problem using Backtracking Algorithm Techniques ", International Journal of Computer Applications, Vol.102, Issue.7, pp. 26-29, 2014.

[10] 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.

[11] 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.

[12] 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.

[13] S.Pothumani, "Solving N-Queens 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.

[14] 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.
[15] Vishal Kesri, Vaibhav Kesri and Prasant Ku. Pattnaik, "A Unique Solution for N-Queens Problem", International Journal of Computer Applications, Vol.43, Issue.12, pp.13-19, 2012.
[16] 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.
[17] 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.
[18] Jun Zhang and Zili Zhang, "An Algebraic method for the n-Queens Problems based on Permutation Operation Group", International Journal of Computer Networks and Information Security, Vol.3, pp.19-25, 2011.
[19] 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.
[20] H. Ahrabian, A. Mirzaei and A.Nowzari-Dalini," A DNA Sticker Algorithm for Solving N-Queens Problem", International Journal of Computer Science and Applications, Vol.5, Issue.3, pp.12-22, 2006.
[21] 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.
[22] Marko Bozikovic, Marin Golub and Leo Budin, "Solving N-Queens Problem Using Global Parallel Genetic Algorithm", European Conference Ljubljana Slovenia, pp.11-17, 2003.
[23] Roc Sosic and Jun Gu, "Fast Search Algorithms for the N-Queens Problem", IEEE Transactions on Systems, Man, and Cybernetics, Vol.21, Issue.6, pp.1572-1576, 1991.
[24] 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.