# Implementation of n-Queens Puzzle using Meta-heuristic algorithm (Cuckoo Search)

Ram Gopal Sharma

*M.Tech Scholar, Suresh Gyan Vihar University, Jaipur, Rajasthan, India*

Bright Keswani

*Head, Department of Computer Applications, Suresh Gyan Vihar University, Jaipur, Rajasthan, India*

**Abstract— This paper use the concept of cuckoo search algorithm for the n-queens puzzle based on the breeding behaviour of cuckoos. We will first introduce the breeding behaviour of cuckoos and the characteristics of L'evy flights and then proposed the algorithm, followed by its implementation. Then, we will compare the proposed algorithm with the backtracking algorithm and compare the different computational results and show the effectiveness of this proposed algorithm. We choose to develop Cuckoo Search application in Java programming language. We used JDK version 7 and NetBeans IDE version 6.8.**

***Keywords*— Meta-heuristic algorithm, cuckoo search algorithm, NP-hard, Constraint satisfaction problem.**

## I. Introduction

Meta-heuristic algorithms are used for combinatorial optimization in which an optimal solution is sought over a discrete search space. The nature-inspired Meta-heuristic algorithms have been used in a wide range of optimization problems, including NP-hard problems such as n-queens puzzle and travelling salesman problem. The n-queens puzzle is the problem of placing n chess queens on an n×n chessboard so that no two queens share the same row, column or diagonal.

Cuckoo search is the Meta-heuristic algorithms. Implementation of cuckoo search in n-queen puzzle involve some assumption such as: position of chess board represent new solution, each queen is considered as egg of cuckoos, and blocks in chess board represent nest of host bird.

The paper proposed an algorithm for n-queens puzzle to eliminate the row, column and diagonal attack and compare with the backtracking algorithm. Finally compare the results and show the effectiveness of cuckoo search algorithm.

The remaining paper is organized as follows: Section II discusses the earlier works done in this field; Section III presents overview of n-queens puzzle; Section IV presents overview of cuckoo search; Section V gives the overview of backtracking Algorithm; Experimental results are discussed in Section VI and Conclusion are presented in section VII. At last References are given in Section VIII.

## II. Related Works

In the past years, so many papers have been published in the field of combinatorial optimization. Meta-heuristic algorithms [3] are used for combinatorial optimization in which an optimal solution is sought over a discrete search –space. X.S. Yang, and Deb S. [1, 4], developed Cuckoo search via L'evy flights in 2009. Isra N. Alkallak [6] developed A Hybrid Algorithm from Cuckoo Search Method with N-Queens Problem in *2012.* Yang, X.S., and Deb, S., [4] developed Engineering Optimization by Cuckoo Search in 2010. In this paper, a Cuckoo search algorithm with improved parameter is used to generate initial population and their fitness is calculated by using objective function. At the end, we find the solution of n-queens puzzle in less time as compared to backtracking algorithm.

## III. N-Queens Puzzle

The n-queens puzzle [7, 8] is the problem of placing n chess queens on an n*n chessboard so that no two queens share the same row, column or diagonal. The puzzle was originally proposed in 1848 by the chess player Max Bezzel as 8-queens puzzle, and over the years, many mathematicians have worked on this puzzle and generalized its n-queens puzzle.

The first solutions were provided by Franz Nauck in 1850. Nauck also extended the puzzle to n-queens problem (on an n×n board—a chessboard of arbitrary size). In 1874, S. Günther proposed a method of finding solutions by using determinants, and J.W.L. Glaisher refined this approach.

N-queen is an NP-hard problem. At first, we need to model the N-queens problem as a constraint satisfaction problem (CSP) [2] problem.

A finite CSP consists of a set of variables associated with finite domains and a set of constraints restricting the values that the variables can simultaneously take. We must find a value for each of the variables that satisfy all of the constraints. A constraint is a relation between local collections of variables.

A finite set of values that each variable can take. This is called the domain of the variable. The Domain of variable Xi is written Di. A solution to a CSP is an assignment of a value from its domain to every variable, in such a way that every constraint is satisfied.

• Variables {Q1, Q2, Q3 ...Qn} represent the queens,

• Domains Q ∈ {1, 2, 3 ...n} ∀ i ∈ {1, 2, 3 ...n}

A constraint satisfaction problem consists of three components
1. Set of variables.
2. Set of values for each of the variables.
3. Set of constraints.

For example, suppose we have a CSP with two variables x and y, and that x can take values {1, 2, 3}, whereas y can take values {2, 3}. Then the constraint that x=y would be written as:

Cxy= {(2, 2), (3, 3)},

and the constraint that x<y would be written as:
Cxy = {(1, 2), (1, 3), (2, 3)}

Between every pair of Variables ($Q_i$, $Q_j$) ($i \neq j$), we have a constraint $C_{ij}$. For each $C_{ij}$, an assignment of values to the Variables Qi=A and Qj=B, satisfies this constraint if and only if
1. $A \neq B$
2. $|A-B| \neq |i-j|$

A solution to the n-queens puzzle will be any assignment of values to the Variables Q1 ...QN that satisfies all of the constraints.

## IV. CUCKOO SEARCH

Cuckoo search [1, 4, 6] is a meta-heuristic optimization algorithm was developed by Xin-She Yang and Suash Deb [4] in 2009. Cuckoo Search was inspired by brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds. Some species such as the ani and Guira cuckoos lay their eggs in communal nests, though they may remove other's eggs to increase the hatching probability of their own eggs. There are three types of brood parasitism: intraspecific brood parasitism, cooperative breeding, and nest takeover. If a host bird discovers the eggs are not their own, they will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere.

Some cuckoo species have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colour and pattern of the eggs of a few chosen host species. This reduces the probability of their eggs being abandoned and thus increases their reproductivity. Generally cuckoo eggs hatch slightly earlier than their host eggs. Once the first cuckoo chick is hatched, the first instinct action it will take is to evict the host eggs by blindly propelling the eggs out of the nest, which increases the cuckoo chick's share of food provided by its host bird. Studies also show that a cuckoo chick can also mimic the call of host chicks to gain access to more feeding opportunity.

Cuckoo Search can be applied for various optimization problems. Cuckoo Search is a reliable approach for embedded system design and design optimization.

A Lévy flight [1] is a random walk in which the step-lengths have a probability distribution that is heavy-tailed. When defined as a walk in a space of dimension greater than one, the steps made are in isotropic random directions. The name "Lévy flight" refers to the French mathematician Paul Pierre Lévy.

Implementation of Cuckoo Search in n-queens puzzle involves some assumption:
1. Positions of chess board represent new solution.
2. Each queen is considered as egg of Cuckoos.

3. Blocks in chess board represent nest of host bird.

Functions used in Cuckoo Search are:

*A. Initial Population:*

In initial population generation n queens are placed on chess board randomly. To enhance the efficiency of the search, the initial population consists of n queen numbered from 1 to n. As we know that only one queen can be placed in one column hence Q[i] queen is placed in $i^{th}$ column which is fixed in whole search process for all i.

Now our task is to find row for each queen. We can represent queen position for $i^{th}$ queen as Q[i]=j, where i represent column value for $i^{th}$ queen and j represent row value for that queen.

In cuckoo search we assume each queen as eggs of cuckoo and chess board position is considered as nest of other birds to put the eggs of cuckoo (queen). Also assume that only one egg can be put in one nest.

Initially put the eggs randomly in nests than find the fitness of the nest to put the eggs in best nest.

*B. Objective Function*

Objective function is also known as heuristic function. In optimization problem in heuristic search or meta-heuristic search our goal is to minimize or maximize the value of objective function. Objective function is used to find out the fitness of candidate solution. There are different heuristic functions for different problems.

In n-queens problem objective function is number of attacks in all over chess board. And our task is two find out the position of chess board in which number of attacks will be exactly zero. Attacks are calculated by calculating number of queens in same row, column or diagonal. Because columns are fixed for each queen i.e. no queen will be in same column, hence we should test only for rows and diagonal. Condition for rows and diagonal attacks are given below:

Condition for rows:

$Q\,i \neq Q\,j \quad \forall i,j \in \{1, 2, .....n\},\ i \neq j$ ……..(1)

Condition for diagonals:

$|\,Qi - Q\,j\,| \neq |\,i - j| \quad \forall\ i, j \in \{1,2,.....n\}\ , i \neq j$ …....(2)

Heuristic h = number of ‘attacks’

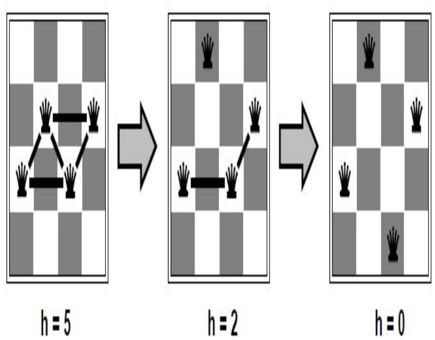


Figure 1:

*C. Proposed Algorithm*

Heuristic or meta-heuristic search process is subjected to generating new solutions after each step. Initial population generated in initialization step is considered as parent node for next generation. In heuristic search techniques generation of new solutions is most considerable step, because most of the time and computation power spends in generating new solutions. If we are able to find out efficient way to generate new solutions, we can reduce computation time and computation power magically.

In implementation of n-queens problem with cuckoo search, we generate new solutions for current best parent node by using following steps;

1. Firstly, for each queen select one random value of row by performing L'evy flight.
2. Calculate the fitness value of each of the solutions using equation 1 and 2.
3. If new value of row for that queen is better than previous best than select it for new generation.
4. Repeat the process until goal node is found or no new generation left

These steps are followed for each iteration of cuckoo search process. After each iteration we get the result better than previous result. This result is considered as parent for next generation.

## V. BACKTRACKING

Backtracking [7, 8] is a general method for finding all or some solutions to some computational problem that incrementally builds candidates to the solutions and abandons each partial candidate c ("backtracks") as soon as it determines that c cannot possibly be completed to a valid solution.

Backtracking is an important tool for solving constraint satisfaction problems, such as crosswords, Sudoku and many other puzzles. Like most recursive algorithms, the execution of a backtracking algorithm can be illustrated using a recursion tree.
The root of the recursion tree corresponds to the original invocation of the algorithm; edges in the tree correspond to recursive calls. A path from the root down to any node shows the history of a partial solution to the *n*-Queens problem, as queens are added to successive rows. The leaves correspond to partial solutions that cannot be extended, either because there is already a queen on every row, or because every position in the next empty row is in the same row, column, or diagonal as an existing queen.

## VI. RESULTS AND DISCUSSION

Our objective in this paper is to compare the results obtained from the Cuckoo search algorithm with improved parameter with the Backtracking algorithm. The programs run on Java programming language on Pentium IV Processor. We used JDK version 7 and NetBeans IDE version 6.8.

A number of experiments have been carried out by giving different inputs and applying Cuckoo Search algorithm and Backtracking algorithm. The results are shown in table 1.The table below shows the comparison of running time between both the algorithms given in seconds.

| S.NO. | Number of Queens (n) | Backtracking | Cuckoo Search |
|---|---|---|---|
| 1. | 10 | 1 | 1 |
| 2. | 15 | 1 | 1 |
| 3. | 25 | 1 | 1 |
| 4. | 28 | 6 | 1 |
| 5. | 30 | 90 | 1 |
| 6. | 50 | - | 1 |
| 7. | 80 | - | 2 |
| 8. | 100 | - | 4 |
| 9. | 150 | - | 17 |
| 10. | 200 | - | 23 |

Table 1: Comparison of Cuckoo Search algorithm and Backtracking algorithm

From the table 1, it is found that using backtracking algorithm we gets the result till n is 29 efficiently and when n is 30 it gives the result in reasonable time but n becomes higher than 30 backtracking algorithm becomes hang due to lot of backtrack. While on the other hand cuckoo search gives result efficiently. So we can say that Cuckoo Search works better than Backtracking algorithm in terms of time taken.

Graphical representation of comparison of results shown below; in this representation x-axis shows the number of queens on the chess board and y-axis shows the computational time required to solve the problem.

Figure 2 shows the difference between backtracking algorithm and cuckoo search algorithm. This difference is due to in backtracking there are too much steps that have to backtrack in search of required result with respect to the size of n. And when n is small enough that does not be seen by user due to high computation power of hardware, but if n is large it requires more steps in backtracking.

While in cuckoo search all the queens are placed in the chess board one queen in each column. Now attacks are only in rows and diagonals. We minimize the attacks by selecting best row for each queen. It can give result instantly for large value of n.

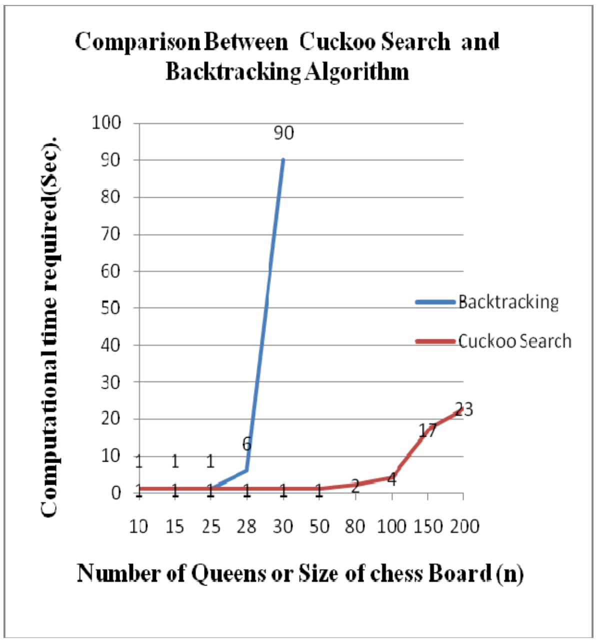


Figure 2: The running time Comparison of Cuckoo Search Algorithm and Backtracking algorithm

## VII. CONCLUSIONS

In this paper we have formulated a new meta-heuristic Cuckoo Search in combination with L'evy flights, based on the breeding strategy of some cuckoo species. The proposed algorithm has been validated and compared with other backtracking algorithm and it shows better results over backtracking for solving n-queens puzzle.

It can be easily extended to study multi-objective optimization applications with various constraints, even to NP-hard problems.

## REFERENCES


[1] X.S. Yang, and Suash Deb., "Cuckoo search via L´evy flights", in: Proceeding. of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009), December 2009, IEEE Publications, pp. 210-214 (2009).

[2] E., Tsang., A Glimpse of Constraint Satisfaction, 1999, Journal Artificial Intelligence review, Kluwer Academic Publishers, vol. 13, pp. 215-217.

[3] X.S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, 2008, pp. 128.

[4] X.S. Yang., and Suash Deb, Engineering Optimization by Cuckoo Search, Int. J. of Mathematical Modeling and Numerical Optimization, Vol. 1, No. 4, 2010, pp. 330–343.

[5] P. Civicioglu and E. Besdok, A conception comparison of the cuckoo search, particle swarm optimization, differential evolution and artificial bee colony algorithms, Artificial Intelligence Review, DOI 10.1007/s10462-011-92760, 6 July (2011).

[6] Isra N. Alkallak, A Hybrid Algorithm from Cuckoo Search Method with N-Queens Problem, *Raf. J. of Comp. & Math's. , Vol. 9, No. 2, 2012.*

[7] Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, Fundamental of Computer algorithms, Galgotia Publications Pvt. Ltd.,2007.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms, Second edition, PHI,2002.