Metaheuristic Techniques on N-Queen problem: DE vs. ABC

A Dissertation II

Submitted

By

Pirzada Nawaz Sharief

To

Department of Computer Science and Engineering

In partial fulfillment of the Requirement for the

Award of the Degree of

Master of Technology in Computer Science

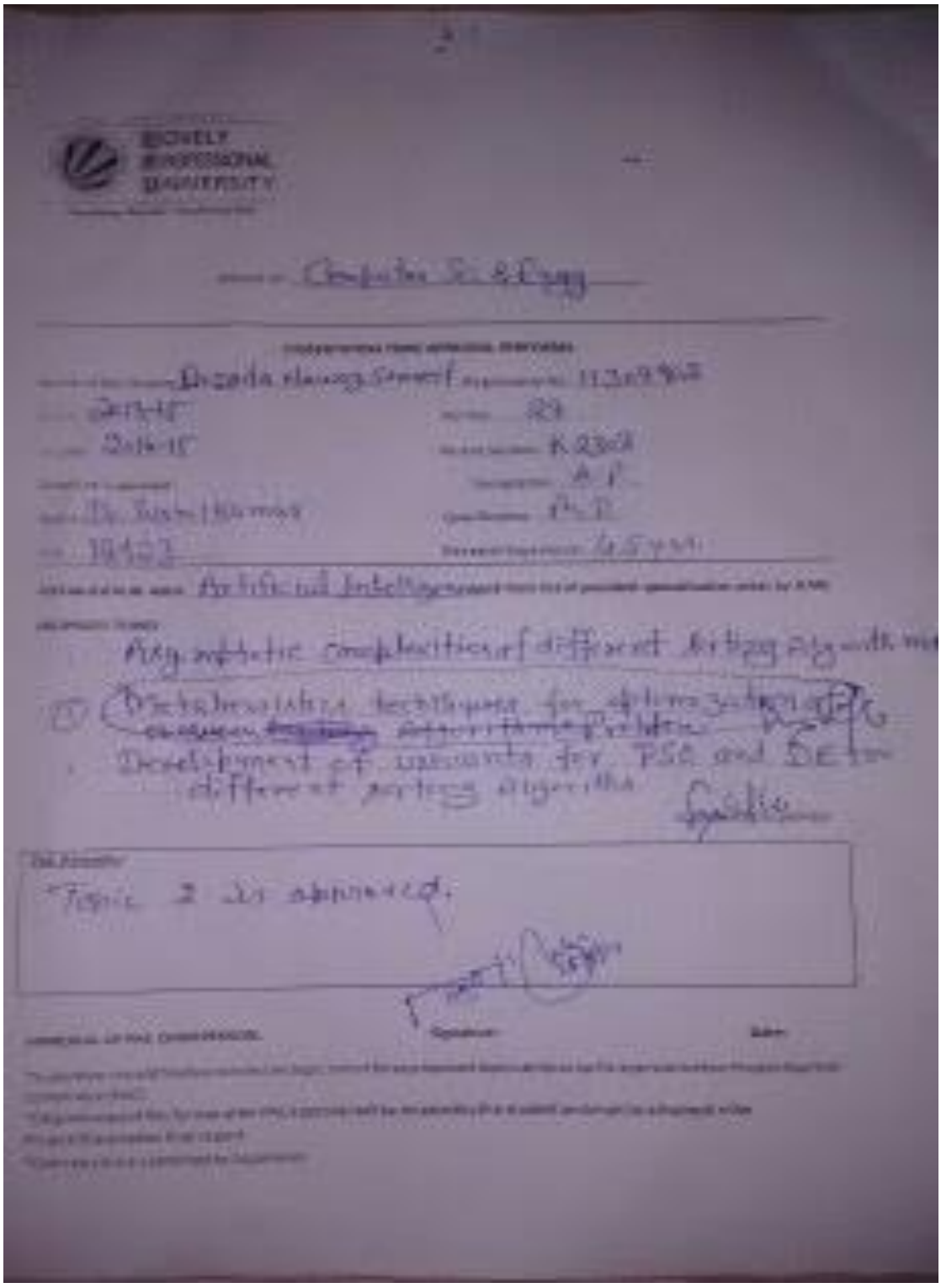
Under the guidance of

Baljit Singh Siani

(Assistant Professor, Lovely Professional University)

(May 2015)

PAC Approval Page



ABSTRACT

Metaheuristic techniques are trending in their implementations on many scientific and real time applications. These techniques include algorithms based on the intelligent evolution or behaviour of humans and animals. Differential evolution being one of such algorithm has found vast usage. The present work details how differential evolution can be optimized for better solutions. The paper proposes to use a mutation strategy that does not require repair function. Also the paper proves that a new crossover over DE enhances the optimization of the algorithm. Conventional artificial bee colony algorithm is also implemented on N-Queen problem. The results are satisfying, indicating DE as better algorithm and ABC as best. Different parameters will be used for evaluation that includes number of function calls (NFCs), success rate (SR) and success performance (SP). The convergence of both algorithms will be measured, on different inputs, from the values NFC. The research will try to find an efficient and fast algorithm for solving many combinatorial problems.

CERTIFICATE

This is to certify that Pirzada Nawaz Sharief has completed M.Tech dissertation proposal titled **Metaheuristic Techniques on N-Queen problem: DE vs. ABC** under my guidance and supervision. To the best of my knowledge, the present work is the result of his original investigation and study. No part of the dissertation proposal has ever been submitted for any other degree or diploma. The dissertation proposal is fit for the submission and the partial fulfilment of the conditions for the award of M.Tech Computer Science & Engineering.

Date:

Name: Baljit Singh Siani
UID: 15359

ACKNOWLEDGEMENT

I would like to present my deepest gratitude to Baljit Singh Siani, Assistant Professor, in the department of Computer Science and Engineering, LPU, for his guidance, advice, understanding and supervision throughout the development of this dissertation study. I would like to thank to the **Project Approval Committee members** for their valuable comments and discussions. I would also like to thank to **Lovely Professional University** for the support on academic studies and letting me involve in this study. I also gratefully acknowledge the support received from Mir Zubair Nazir and Junaid Nazir throughout this research.

(Pirzada Nawaz Sharief)

DECLARATION

I hereby declare that the dissertation proposal entitled **Metaheuristic Techniques on N-Queen problem: DE vs. ABC** for the M.Tech degree is entirely my original work and all the ideas and references have been dully acknowledged. It does not contain any work for the award of any other degree or diploma.

Date: _____

Investigator: Pirzada Nawaz Sharief

Registration. No. 1130890

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Evolutionary algorithms are the methods inspired by physical process, natural evolution and stochastic events. The prime advantage of these methods is their flexible model due to which these can be applied to a wide spectrum of problems. The popular meta-heuristic algorithms include Evolutionary Algorithms (EA) Genetic Algorithms (GA) [10], Ant Colony Optimization (ACO) [11], Particle Swarm Optimization (PSO) [12], Artificial Bee Colony (ABC) [5][6][8], Differential Evolution (DE) [1][2][[3][7] etc. The meta-heuristic focused in the present work is Differential Evolution (DE), an evolutionary algorithm proposed by Storn and Price (1995) [7] and Artificial bee colony algorithm, proposed by Karaboga (2005) [8].

1.1 EVOLUTIONARY ALGORITHMS

Evolutionary algorithms are part of derivative free optimization and search methods like simulated annealing (SA), random search, and downhill simplex search and consist of evolutionary strategies, evolutionary programming, genetic algorithms and genetic programming. Due to their adaptive behaviour evolutionary algorithms are capable of solving non-linear and high dimensional problems without the requirement of differentiability or the explicit knowledge of problem structure.

The two basic driving forces behind the evolutionary systems are

1. Variation operators (Recombination and Mutation): This is responsible for creating the assortment and thus leads to innovation.

2. Selection that controls quality.

The mutation holds the important step in the evolutionary process. The general mutation strategy involves deriving a solution from randomly selected solutions. This ensures the probabilistic concept of exploring the search space. The difficulty faced here is that there are certain redundant traits that get pass into the offspring. For this a repair function is needed which ensures that the offspring does not hold any redundant traits. The crossover is responsible for recombination of parents to produce their offspring. No other individual of the population is involved in this process except parents. Next comes

the selection, which is the process of selection of best individuals out of mutation and target solutions.

Explaining the two in the context of general Scheme of evolutionary algorithm in pseudo code fashion as:

START
COMPUTE the population.
EVALUATE each member of the population.
REPEAT (the terminating condition is met) DO
    SELECT members from the population as parents.
    RECOMBINE pairs of parents selected.
    MUTATE the resulting candidate.
    EVALUATE new candidates
    SELECT individuals for the next generation.
END

The compute step generates a random population of N candidates. These candidates are represented in different manner in different types of algorithms. In Evolutionary algorithms these candidates are represented as real valued vectors. In genetic algorithm they are represented as strings, as trees in genetic programming etc.

In selection the individuals which will produce the offspring are chosen. It starts with the fitness assignment. Every individual in the population pool receives a reproduction probability which depends entirely on its own objective value and off all the others. This fitness is then useful for afterward selection. The recombination and the mutation form the basic operation for variation. Recombination generates new candidates by combining the information present in two or more parents/candidates (parents - mating population). This is accomplished by combining the variable values of the parents. Different methods must be used depending on the representation of variables in parent candidates.

In mutation individuals are aimlessly altered. The variations are predominantly small. They will be applied to the variables of the individuals with a low probability (mutation probability or mutation rate). It's after recombination that offsprings are mutated and finally selection of best of original and mutation output is done.

1.2 THE TEST-BED, N-QUEEN PROBLEM

Initially proposed as a chess puzzle, the 8-queen problem originated in the mid-1880s is one of the classical combinatorial problem that has been used as a test bench to compare and analyse the performance of many problem solving algorithms. The problem is to place N-Queens on N by N chessboard in a manner that no Queen attacks any other Queen placed on the chessboard. Since a Queen has a property of moving in any of the directions, two horizontal, two vertical and four diagonal, thus finding its place in NP-Complete problem set. This means that any algorithm that works efficiently on this problem will have the ability to adapt to likewise problems with ease. The task here more than finding a solution is to compare two optimization algorithms, artificial bee colony algorithm and modified-DE.

The reason behind choosing N-Queen problem is that it turns to be computationally very expensive. Consider a case where N is 8, we will be having 4,426,165,368 possible arrangements of 8-Queens form which only 92 solutions are valid. The brute force technique in 8-Queen case differentiates the solution space to 40,320, which can be thought of manageable space but with greater N it becomes very inefficient. It's possible to reduce this complexity using brute force technique but then we are just reducing the number of possibilities to 16,777,216 and then again applying the same technique reduces it to 40,320. This technique is computationally manageable for only n=8 but would be infeasible greater values of n. So the scope was there to try solve the n-queen problem using heuristic techniques.

1.3 DIFFERENTIAL EVOLUTION

Differential evolution starts with the generation of random vectors together called as population. The vectors are generated as:

$$x_i^G = x_{i(L)} + rand_i[0,1].\left(x_{i(H)} - x_{i(L)}\right) \tag{1.1}$$

Where $x_{i(L)}$ and $x_{i(H)}$ are the lower and higher boundaries of dimensional vector $x_i = \{x_{j,i}\} = \{x_{1,i}, x_{2,i}, ..., x_{d,i}\}^T$.

Next is the Mutation step. For each parent parameter vector, DE generates a candidate child vector based on the distance of two other parameter vectors. For each dimension $j \in [1, d]$, this process is shown, as is referred to as scheme DE/bin/1 by Storn and Price.

$$x' = x_{r3}^{G} + F.(x_{r1}^{G} - x_{r3}^{G}) \tag{1.2}$$

Where, the random integers $r_1 \neq r_2 \neq r_3 \neq i$ are used as indices to index the current parent object vector. As a result, the population size N must be greater than 3. F is a real constant positive scaling factor and normally $F \in (0,1+)$ controls the scale of the differential variation $\left(x_{r1}^{G} - x_{r2}^{G}\right)$.

Crossover is the process of generation of the trial vector from the original population vector and the mutant vector. This is accomplished by shuffling the competing vectors. Selection is the last step in Differential evolution algorithm. It decides which vector trial vector or the original candidate vector to be the next generation member. It is based on the greedy approach, for the minimization problem a lower value vector is chosen. The whole cycle of evolution is repeated till the termination criterion is satisfied. The detailed working of Differential Evolution algorithm is as:

The Differential Evolution was initially proposed in 1990s to optimize the problems with continuous variables [1], [2], but has now found its application in solving many combinatorial problems [8], [9], [13]. The two basic driving forces behind the Differential Evolution algorithm are variation operators which include recombination and mutation, responsible for creating assortment and thus leading to innovation and the selection that controls quality. Being a population-based directed search method [12], its working is like other evolutionary algorithms and follows a sequence of steps to converge to a solution. The DE starts with the random initialization of the population vector. The initialization is done as:

$$x_i^{G} = x_{i(L)} + rand_i[0,1].\left(x_{i(H)} - x_{i(L)}\right) \tag{1.3}$$

Where $x_{i(L)}$ and $x_{i(H)}$ are the lower and higher boundaries of dimensional vector $x_i = \{x_{j,i}\} = \{x_{1,i}, x_{2,i}, ..., x_{d,i}\}^T$. $rand_i[0, 1]$ is a random number generated in (0, 1) [17]. The initialization is done only a single time, at the start of algorithm but the rest of the

processes like mutation, crossover and selection run in every epoch till the convergence. Every epoch is also called as a single generation.

Mutation - The mutation step generates a corresponding mutant vector, also called as donor vector for every $x_{i,G}$ in generation G.

$$v_{i,G} = x_{r1}^{G} + F.(x_{r2}^{G} - x_{r3}^{G}) \quad (1.4)$$

Where, the random integers $x_{r3}^{G}, x_{r1}^{G}, x_{r}^{G}$ are randomly chosen vectors and r1≠r2≠r3≠i. The value of is called the mutation factor. The higher value of F will result into higher diversity in the population and lower value of F will lead to faster convergence. Different mutation strategies for differential evolution are as:

DE/rand/1 $v_{i,G} = x_{r1}^{G} + F.(x_{r2}^{G} - x_{r3}^{G})$ (1.5)

DE/rand/2 $v_{i,G} = x_{r1}^{G} + F.(x_{r2}^{G} - x_{r3}^{G}) + F.(x_{r4}^{G} - x_{r5}^{G})$ (1.6)

DE/best/1 $v_{i,G} = x_{best}^{G} + F.(x_{r2}^{G} - x_{r3}^{G})$ (1.7)

DE/best/2 $v_{i,G} = x_{best}^{G} + F.(x_{r2}^{G} - x_{r3}^{G}) + F.(x_{r4}^{G} - x_{r5}^{G})$ (1.8)

DE/ran-to-best/1 $v_{i,G} = x_{r1}^{G} + F.(x_{best}^{G} - x_{r3}^{G}) + F.(x_{r4}^{G} - x_{r5}^{G})$ (1.9)

Crossover - The crossover is responsible for increasing the diversity of the population. The binary crossover is used for classical DE (DE/rand/1/bin). The vector generated here is called the trial vector. The binary method of crossover is defined as:

$$U_{i,G} = (U_{1i,G}, U_{2i,G,.....} U_{Di,G}) \quad (1.10)$$

$$U_{ji,G} = \begin{cases} v_{ji,G} & if\ rand_j(0,1) \leq Crossover\_rate \vee j=k \\ x_{ji,G} & otherwise \end{cases}$$

Crossover_rate ∈ (0, 1) is the crossover rate that is predefined. k ∈ {1,2,3,…..,D} , taking the value of k as selector in crossover makes sure that at least on dimension is always selected form the mutant vector.

Selection - Selection is the last step in the iterative process of the Differential Evolution. The selection choses a single vector between $U_{i,G}$ and $x_{i,G}$ for the next generation on the basis of fitness value. For minimization problem the selection chooses a vector which

has minimum fitness value and for maximization problem it chooses the one with maximum fitness value. This is called the greedy selection.

1.4 SWARM INTELLIGENCE

"The emergent collective intelligence of groups of simple agents".

Swarm intelligence is a system property where global patterns emerge due to combined behaviour of agents interacting locally with the environment. The properties associated with Swarm intelligent systems are:

1. The system is based on a flock of individuals.
2. The individuals that are homogenous form the system, homogenous in sense that they are all either identical or may be of few topologies.
3. The individual interactions are entirely based on simplistic behavioural rules which in-turn use only local information that population individuals exchange directly or through the environment.
4. The final behaviour of the system is the result of individual interactions and the interaction of individuals with the environment.

Research in swarm intelligence can be of following types.

1. Natural vs. Artificial: - the study of intelligence of different biological systems come under natural swarm intelligence research and the human artifacts research comes under artificial swarm intelligence research.
2. Scientific vs. Engineering: - This classification is totally based on the set goals. The scientific research is based on simulating the swarm intelligent models and study the various kinds of relations that exist in such swarm intelligent systems. The engineering aspect of the swarm intelligence is to study the different swarm intelligent models and apply them to solve different section of problems.

Some examples of systems that are being studied swarm intelligent system are.

1) Ant and termite colonies
2) Schools of fish.
3) Bird flocks
4) Herds of land and animals.
5) Foraging behaviour of bees.

Besides above the applications that fall in swarm intelligence are some human artifacts, some multi-robot systems and certain computer programs that are meant to tackle optimization and data analysis problems. Based on some reaction-diffusion equations, Tereshko came up with a model explaining the foraging behaviour of honey bees. The model consists food Sources, Employed foragers, and Unemployed foragers as its main components and defines the primary behaviour of honey bee colony in Selection of food source and abandonment of food sources.

Food Sources: - Selection of food source depends on several properties associated with a particular food source which include the location of the food source, energy content and the ease or the difficulty faced in extracting the nectar. But to keep it simple the quality of the food sources defines the main constraint for the selection of food source.

Employed Foragers: - The employed foragers are the ones being employed by the onlookers on some specific food sources. The onlookers waiting in the hive receive the information about the food source from the employed forager which the later gathers from the food source.

Unemployed Foragers: - Unemployed forager can be either a scout or an onlooker, both trying to exploit the food source but by different means. The onlooker does so by interpreting the information that it gets from the employed bees and the scout does it by randomly searching the environment.

The most paramount occurrence is the exchange of information among bees which leads to the formation of combined erudition or knowledge. The dancing area holds the most crucial area with respect to the process of information exchange. The information about the quality of food sources is passed in the dancing area. Waggle dance is the name given to the dance that happens while communication. The onlookers always have the choice of placing themselves at the most profitable source. The selection of the food source by onlookers depends on the profitable returns from the food source. The value of probability with which a particular food source gets selected is passed itself by the employed bees to the onlookers and it takes longer time if the communication is through waggle dance. Examine the fig 1, let’s assume A and B as the two discovered food sources. At first there are two possibilities for an unemployed forager bee.

1. It can immediately search for the food source due to its internal intuition which makes it a scout.

2. It starts searching after watching the waggle dance which makes it a recruit (R in the fig. 1).
3. Once the food source is found the bee crams it and then automatically starts exploiting it, making the bee an employed one. The bee the transports the nectar which it extracts from the food source to the hive, where it unloads it. The whole process is followed by three options for the bee.
    a. UF- Uncommitted follower post abandoning the food source.
    b. EF1- Before returning to the food source it might have a dance at the hive and recruit some more nest mates.
    c. EF2- The state where it doesn't recruit and continues its foraging.

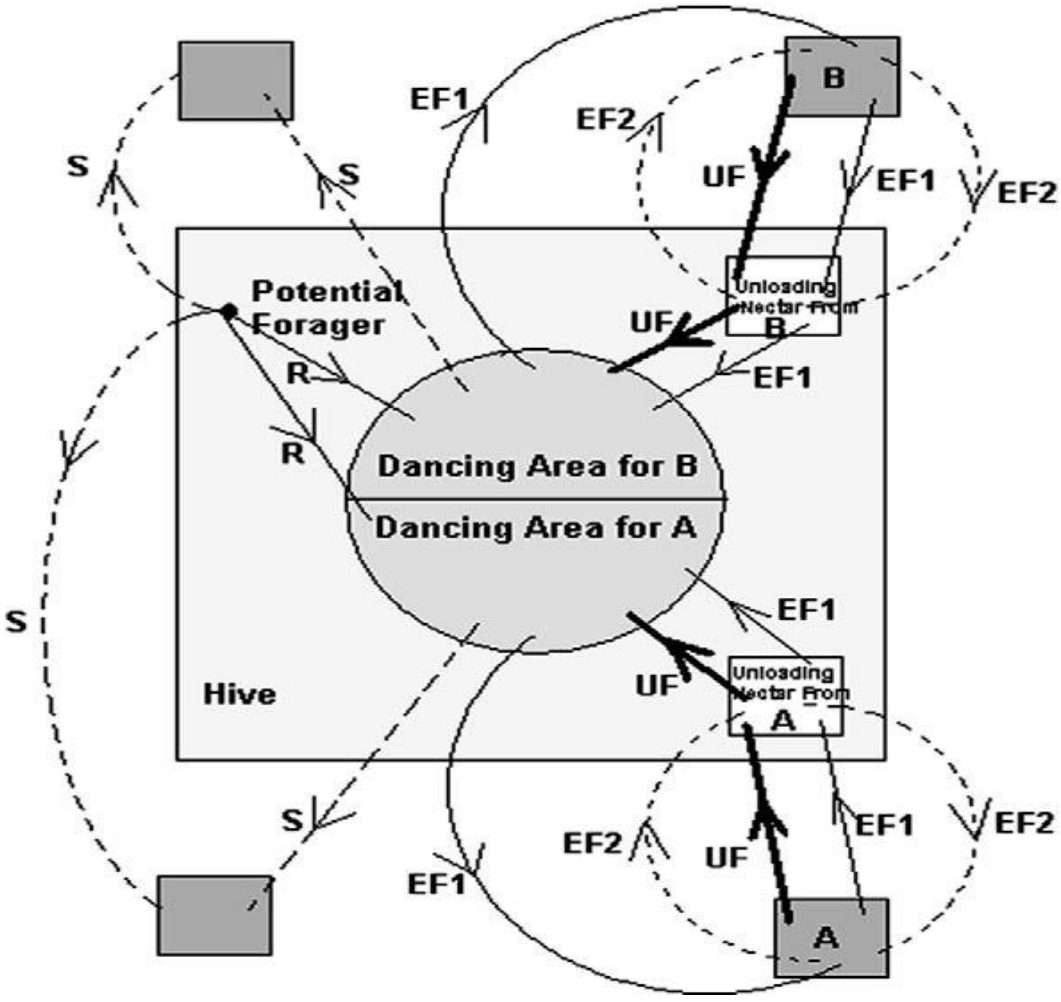


Figure 1: Representation of Foraging Behaviour of Bees

A point of highlight is that all the bees doesn't forage at once. The conclusion has been drawn from different experimentations that the bees forage based on a rate that is proportional to the difference of presently foraging bees and the total number of bees present.

1.5 ARTIFICIAL BEE COLONY ALGORITHM

Artificial Bee Colony algorithm commonly known as ABC algorithm was proposed by Karaboga in 2005 as an optimization algorithm. In ABC algorithm three kinds of bees exist. Employed bees, onlooker bees and the scouts. An onlooker decides the food source and keeps waiting for the same at the dance area, scout bee remain in the task of discovering new food resources whereas the employed ones keeps going to the food source visited by it before. The site of a food source represents a probable solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the related solution, calculated by:

$$fit_i = \frac{1}{1 + fit_i} \tag{1.11}$$

The onlookers select the food source according to the probability value calculated as:

$$P_i = fit_i / \sum_{1}^{N} fit_n \tag{1.12}$$

Where N represents the total number of food sources which equals the number of employed bees or onlooker bees. A new candidate position is produced from the old one in memory using the following expression:

$$V_{i,j} = Z_{i,j} + \varphi_{i,j}\left(Z_{i,j} - Z_{k,j}\right) \tag{1.13}$$

Where $k = \{1, 2. . . SN\}$ and $j = \{1, 2. . . D\}$ are aimlessly chosen indexes. Although k is determined randomly, it has to be different from i. $\varphi_{i,j}$ is a random number between [−1, 1]. It pedals the creation of neighbour food sources around $Z_{i,j}$ and represents the assessment of two food positions noticeable to a bee. The new food source is selected over the previous one if the latter's nectar is abandoned by bees. In ABC, this is replicated by producing a position aimlessly and replacing it with the neglected one. In ABC, when a position of food source can't be improved through predetermined number of cycles (PNC), then it is considered as the abandoned one. The value of PNC is a main

control parameter of the ABC algorithm, called as “limit” for abandonment. Assuming Z_i as the abandoned source and a new food source is discovered by scout, then

$$Z_i^j = Z_{\min}^j + random(0,1)\left(Z_{\max}^j - Z_{\min}^j\right) \tag{1.14}$$

The performance of each $V_{i,j}$ produced is checked with the old one and if the new food source matches or has higher nectar than the old one, it’s replaced with the old one in memory else not. A greedy selection operation takes place. The control parameters of ABC are 1) SN, the number of food sources 2) The value of limit and 3) the MCN, maximum cycle number. The generalization of the artificial bee colony algorithm can done as:

1. Initialize the population with random food sources.
2. Repeat
3. The employed bees are placed on their food sources.
4. The onlookers are placed on the food sources on basis of nectar amounts associated to food sources.
5. The discovery of food sources takes place by the scouts.
6. Remembering the best source found so far.
7. Until the convergence is met.

CHAPTER 2

LITERATURE REVIEW

This survey examines the current literature pertaining to the analysis of differential evolution and artificial bee colony algorithm in solving different combinatorial problems with N-queen problem being the case of concern in our study. Although the implementation of artificial bee colony on N-queen problem is yet to be done, this survey will present ABC as an optimization algorithm on other problems.

Shahryar Rahnamayan and Paul Dieras [1] compares two evolutionary algorithms DE and CMA-ES on solving N-queen combinatorial problem. The main measures being convergence, velocity and robustness. Three metrics, success rate (SR), number of function calls (NFC) and success performance (SP) being used to compare the algorithms. The experiment is performed on 12 trials starting with 4-queeen problem to 15-queen problem. The experimental results show that CMA-ES outperforms DE in 4 cases (for N= 4, 5, 8, 9). DE performs better than CMA-ES in nine other cases. Despite this CMA-ES shows higher robustness than DE due to DE having lower success rate of 0.49 and 0.90 for CMA-ES. The conclusion drawn was DE performs better that CMA-ES on separable functions while as in case of non-separable functions CMA-ES outperformed DE.

Ricardo S.Prado et al. [2], proposed an approach for optimization of combinatorial problems is proposed using DE algorithm. This aims at saving its interesting mechanism of search for discrete domains, by defining the variation between two candidates as a differential list of movements in search space, leading to more common differential mutation operator. Three alternatives have been used for using the differential list of movement within the mutation vector operation. The results are produced on instances of travelling salesman problem and N-Queen problem. The goal of this paper is not to produce a new metaheuristic for TSP or N-queen rather it's to compare the differential mutation mechanism.

Proposed differential list of movements

Definition 1:- The application of valid movements m_k of differential list of movements $M_{j \to i}$ to a solution $s_j \in S$ leads to a solution $s_i \in S$.

$$M_{j \to i} = s_i \ominus s_j \tag{2.1}$$

Where $\ominus$ returns a list of movements. The application of $M_{j \to i}$ to s_i is defined as:

$$s_i^{'} = s_i \oplus M_{a \to b} \tag{2.2}$$

Where the operator $\oplus$ receives an applicable solution and a list of movements, returning other solution

$$s_i = s_j \oplus M_{j \to i} = s_j \oplus (s_i \ominus s_j) \tag{2.3}$$

The multiplication of differential list of movements with constant F has also been defined.

Definition 2:- The $F \otimes M_{i \to j}$ returns a list comprising of first $[F * |M_{i \to j}|]$ movements from $M_{i \to j}$. F is a constant $F \in [0,1]$ and $|M_{i \to j}|$ is the size of list.

Definition 3:- The $F \otimes M_{i \to j}$ returns a list of movements with probability F selected from $M_{i \to j}$.

Definition 4:- The $F \otimes M_{i \to j}$ returns a list comprising of randomly chosen $[F * |M_{i \to j}|]$ movements from $M_{i \to j}$

Analysis of results:

The proposed method was evaluated against the RPI (Relative position indexing approach) and PM (Permutation matrix) methods and was able to achieve slightly better solutions. RPI came as the worst in terms of convergence speed.

The results presented the differential list approach performing better or equal to other approaches like RPI and PM with following advantages.

1. Doesn't require repairing operators
2. The quality of being general. Can be applied to other combinatorial problems as well.
3. The property of being flexible
4. For discrete domains, preserving the principles and search mechanism.

The problem with this approach is that it's not certain which definition would optimize the performance in which combinatorial problem. Here the definition 2 provided better

results with TSP but not for N-queen which produced better results with definition 4. This requires further investigation. I may frame my problem as

The problem would be to check DE using all the definitions against the latest swarm intelligent algorithm, Artificial Bee Colony and evaluate the results.

The authors Brain Hegerty et al. [3] compare the performance of differential evolution and genetic algorithm. The comparison criteria are 1) convergence speed 2) computational complexity 3) accuracy 4) stability. The paper concludes at a point establishing differential evolution as much more robust. In both problem spaces the paper considers (Travelling salesman problem and N-queen problem) differential evolution's results proved to be much more valuable than genetic algorithm. In TSP quick results were produced by both algorithms, but differential evolution didn't stop to improve on the city tour, where the GA could be seen to freeze in non-optimum solution with higher frequency. As the population is increased it effects the iteration time for a generation. With larger values of N the GA proved to be efficient in terms of running time than DE on some cases. This is due to the overhead caused due to computational complexity of DE. The experimental results show that for larger populations DE outperforms GA in both run time and number of generations.

The fact derived presents DE approach as better solution than GA for combinatorial problems. So DE turning out to be the best solutions, the problem is to find out if it stands against the latest swarm intelligent algorithm, Particle swarm optimization.

Abdul GhaniAbro and Junita Mohamed-Saleh[4] explains the artificial bee colony algorithm in terms of possibility of generation of poor solutions and provides a remedy to it by proposing a new solution in form of new scout-bee stage of ABC algorithm. This method capitalizes on so-far the best-found possible solution. The new solution is evaluated against original ABC algorithm and the results mark the need to enhance the ABC algorithm for best possible results.

In original ABC the scout bee does the work of replacing the abandoned food source. The equation used is

$$y_{ij} = y_j^{\min} + rand(0,1)\left(y_j^{\max} - y_j^{\min}\right) \tag{2.4}$$

Where $y_j^{\min}$ the lower bound of search is space and $y_j^{\max}$ is the upper bound of search space. As evident from the equation the food source is randomly chosen making

chances of picking a fitter-food source very thin. The proposed solution picks up the best source rather than the random one. The proposed equation is given as

$$z_{nj} = \left(y_{best,j}\right) * \beta_{nj} \tag{2.5}$$

z_{nj} is the jth dimension-magnitude of the recently assigned food source and $y_{best,j}$ is the jth dimension-magnitude of the global best of the global-best food source and β_{nj} is the random number within [0.90, 1.10]. The results were compared on the convergence parameter. The enhanced ABC showed much better convergence than the basic ABC algorithm. Although the Enhanced ABC showed better results than ABC in convergence rate, it will be an interesting research to evaluate the earlier against other evolutionary algorithms on any combinatorial problem like N-Queen.

The Li-Pei Wong et al. [5] uses the bee colony optimization algorithm to solve Travelling salesman problem. The experimental results obtained are compared with other existing approaches on set of benchmark problems. The performance is investigated on some problems form TSPLIB. The dimensions of the problem lie in the range of 48 to 318 cities. The results depict that BCO achieves good performance on some of the benchmark problems. The results also show that there is a scope of improvement in the algorithm when considered in respect of other approaches. The weapon of achieving optimality here is that these approaches use local search techniques. This remains its future work to incorporate these local search techniques to improve its performance.

Dervis Karaboga and Bahriye Akay [6] evaluates the performance ABC algorithm against the standard versions of DE, PSO, GA and ES optimization algorithms. The work uses ABC for optimizing a big set of numerical test functions and then compares the results with those generated by the standard versions of genetic algorithm, particle swarm optimization, differential evolution and evolution strategies. The ABC performance was seen to be similar or better than others with the main advantage of employing only fewer control parameters. Few experiments were conducted manipulating a single step in each experiment. The modified versions were also provided but with much higher complexity than the original one.

Shima Sabet, Fardad Farokhi and Mohammad Shokouhifar [24] present a hybrid artificial bee colony algorithm on instance of travelling salesman problem. Since

travelling salesman problem is NP-complete several evolutionary algorithms have solved the problem. The paper applies a new swarm optimization algorithm on TSP that comes with the mechanism which tries to find new better solutions around the previous ones. The authors present a new hybrid mutation mechanism for artificial bee colony algorithm that performs search for neighbour solutions for the bees, in case of travelling salesman problem. The experimental results showed that the modified approach was able to find the minimum cost path solutions very quickly.

Musarat Ali, Millie Pant and Ajith Abraham [25] shows the effect of generating the initial population in differential evolution algorithm without the conventional methods that include generation with computer generated random numbers or quasi random sequences. The selection of the initial population in heuristic optimization algorithm holds much importance as it affects the initial search along which the solution is tried to found. Also the initial population has a good influence on the final solution. A non-linear method is used in conjunction with the conventional random number method to generate the initial population of the differential evolution. The algorithm proposed was named as NSDE and a test on 20 benchmark problems proved this new method of generating initial population much efficient than the conventional method. The results show the scheme used improves the performance of differential evolution algorithm in terms of converge rate and average CPU time.

Dervis Karaboga and Bahriye Basturk [26] proposes a new optimization technique based on the foraging behaviour of bees. They define swarm intelligence in terms of ant colony optimization, bee colony optimization etc. The paper explains the swarm intelligence in bees in their behaviour of swarming around the hive. In this paper the authors use their proposed artificial bee colony optimization algorithm for optimizing the many variable functions. The results are compared with those generated by algorithms like genetic algorithm, particle swarm optimization algorithm, particle swarm inspired evolutionary algorithm and the results of ABC outperformed the other algorithms. The algorithms were applied on five high dimensional numerical benchmark functions.

Ming-Huwi Horng [reference 4] finds an implementation of artificial bee colony algorithm on image processing. The paper proposes a novel multi-level algorithm namely MEABCT based on the already present artificial bee colony algorithm. MEABCT, maximum entropy based artificial bee colony thresholding is an MET

algorithm. The MEABCT comes up with an algorithm resulted from the simulation of foraging behaviour of bees, used for the selection of appropriate threshold values in problems of image segmentation. The results derived from this modified implementation are promising and can be used to solve much more complex image problems like complex document analysis. **Sinha, S.N et.al [14]** use the differential evolution algorithm to present a cryptanalytic attack on Merkle-Hellman knapsack cipher. Differential evolution being a stochastic optimization algorithm has three operators namely mutation, crossover and selection to look for the solution in the solution space. The results when compared with those of other techniques like genetic algorithms for cryptanalysis of knapsack cipher proved the former as the best of the techniques for this purpose.

Turkey, A .M. et.al. [15] Tries to solve the larger N-valued n queen problem with the genetic algorithm. Since N-Queen problem is a well-known classical NP-Hard combinatorial problem, solutions to this problem on small N values can be found by linear programming or any classical search algorithms can be found in accepted time value. The paper presents N-queen problem as CSP, constraint satisfaction problem, defines it to be time consuming problem on larger N-values. The paper adopts a genetic algorithm to solve this combinatorial problem. The results that the evolutionary approach of genetic algorithm as the best approach to solve such types of problems. Thus genetic algorithm comes as the efficient technique and is able to produce good results on such types of problems.

Tuner, J and Ali, S. [16] uses genetic algorithm to solve up to 200-Queen problem. The value of N being less or equal to 200. The paper shows how the operators of genetic algorithm including mutation, crossover and selection are efficient in handling the complexity issues of combinatorial and constraint satisfaction problems such as N Queen. Other same types of problem include travelling salesman problem, knapsack problem etc. The genetic algorithm finds a solution to the problem by treating it as sequencing or ordering problem and without any knowledge sees through the search space satisfying many constraints posed by the acquired complexity of the problem.

Rok Sosic et.al. [17] Presents a case study on N-Queen problem where they use a new efficient search that comes with minimum conflicts. The backtracking solution to the combinatorial problems is not counted as the efficient technique due to the problem that its computational time grows exponentially. An alternate solution to the problem is

given that eliminates the problems of backtracking that comes up in solution convergence. A new search technique is proposed against the backtracking search, namely, local search with conflict minimization. The proposed technique is applies on one of the most complex classical combinatorial problem, N-Queen problem. An efficient local search algorithm applied on N-Queen does not backtrack and the algorithm runs in the linear time. The method has the capability to solve much larger size N-Queen problems. A workstation using this algorithm can solve 3000000 queen problem in less than 55 seconds.

Heris, J.E.A et.al. [18] Make an attempt to modify a genetic algorithm for solving N-Queen problem. The paper defines holism and random choices as the root of the problem in searching large state space, for genetic algorithms. The paper explains the drop in the efficiency of genetic algorithms when the size of solution space to be searches grows exponentially. A solution is provided for this very problem by using a local search algorithm. The paper uses minimal conflicts algorithm as local search algorithm. It attempts to locally search a solution or state space. The implications of using this may cause the more or longer number of steps for the GA to arrive at the solution. So the modified version uses the blend of genetic algorithm and local search algorithm. When the performance of the genetic algorithm is compared with the modified version the modified version of the algorithm comes out be the winner on the performance measure.

Xiano-Hing Hu et.al. [19] Presents a review on ripple spreading genetic algorithm for many combinatorial problems. The permutations based implementations of the genetic algorithms come up with certain problems like the generation of infeasible solutions by many evolutionary operations, the memory wastage and the restrain to scalability of the algorithms and inability to application of classical binary operators without prior modification. The issues are addressed in this paper with presentation of new scheme for the application of binary based genetic algorithms to combinatorial problems. The scheme is based on the ripple –spreading model. The scheme is named as ripple spreading genetic algorithm (RGSA). Based on the previous implementations of RGSA, this work presents a collective and comprehensive review of the procedure of RGSA. The paper also presents the analysis of different new technologies that could be used in RGSA for further optimization.

Yu Liang et.al. [20] Modifies an existing artificial bee colony algorithm and comes with modified artificial bee colony algorithm meant for large scale optimization. The paper reviews the artificial bee colony algorithm as very good in solving and optimizing combinatorial and numerical function problems but also mentions its lag in optimizing problems with higher dimensions. Cooperative coevolution is the mechanism embedded in ABC which benefits with the discovery of relations among high dimensional variables. The modified ABC instead of optimizing the single dimension optimizes a single group, comprising of the many relationship dimensions. The modified algorithm is tested on very large scale optimization benchmarks and the performance is compared with the original artificial bee colony algorithm (ABC). The results are also evaluated against two classical CC frameworks CCVIL and DECC-G and CCABC was proved to solve large scale optimization problems efficiently.

Abu-Mouti, F.S. et.al. [21] Overviews the karaboga's artificial bee colony algorithm and its practical applications. The ABC being introduced as meta-heuristic technique based on random initialization of population is inspired from the foraging behaviour of bees. The ABC proves to be efficient in attaining local solution or divergence. This is of much use in real world problems which have very high degree of uncertainty and which are difficult to solve with the earlier conventional algorithms. The paper discusses the prominent features of the artificial bee colony algorithm, performance against other algorithms and its characteristics. The ABC proves to be is competent in outperforming other algorithms, especially on highly complex problems.

Babayigit, B. et.al [22] presents artificial bee colony with newly defined inversely proportional mutation strategy. It solves the numerical function optimization problems much efficiently than original ABC. The method is intended to enhance the exploitation capability of the original artificial bee colony algorithm. The experimentation on nine famously used benchmark problems show that modified ABC with modified inversely proportional mutation outperforms the standard ABC algorithm.

Martinjak, I. et.al. [23] Attempts to compare meta-heuristic algorithms while solving N-Queen problem. The paper presents the manner in which different heuristic algorithms can be used to solve N-Queen combinatorial problem. Metaheuristic techniques for tabu search, simulated annealing and genetic algorithm are shown. Test results are analysed and the complexity on the upper bound is determined. Comparison of efficiencies of different algorithms and their achievements are measured. The paper

shows that the fitness function complexity can be differentiated to O (1) which solves high dimensional problems easily.

CHAPTER 3

PRESENT WORK

3.1 Problem Formulation

Metaheuristic techniques are the trending topics in computing world committed to research and analysis. It started with the simulation of evolutionary process of humans into programming concepts to find solutions of many complex problems that were otherwise hard or impossible to solve. Genetic algorithm was one of the first that came as the evolutionary algorithm with operators that resembled the different mechanisms humans go through such as mutation, crossover and selection. Over the time the need arrived to modify the algorithms or find the new ones. The modification was needed so as to solve more complex problems efficiently or to have at least a solution. This resulted the arrival of different evolutionary algorithms like differential evolution, CMA-ES, Genetic programming, evolutionary programming etc. These algorithms come with the same underlying idea: with the given population of individuals the survival of fittest applies due to natural selection and this enhances the fitness of the population. The next generation is chosen based on the fitness value, the better the fitness value of candidate solution, more is the probability of it being selected for the next generation. This process of selection is proceeded by process like mutation and crossover. The whole process is iterated until a solution of desired fitness is arrived. This whole process of evolutionary algorithms has been proved beneficial in deriving the solution of many real time applications and other problems as well. One of the section that these algorithms has found impressive implementation are combinatorial problems and constraint satisfaction problem. One of the classical complex combinatorial problem is N-Queen problem found by famous chess player Max Bezzel. The problem is expressed as:

Placement of N Queens on N by N chessboard in a way that no queen on the board attacks any other queen.

The results by Shahryar Rahnamayan et.al. [1], in solving N-Queen with differential evolution looked like the differential needed some improvement so as to increase the convergence rate of the algorithm, so more than solving N-Queen with differential evolution algorithm the problem here is “ Improve the classical differential evolution

algorithm for faster convergence rate". The tests done here on N-Queen combinatorial problem.

Swarm intelligence came with the concept of mapping the intelligent behaviours of animals into algorithms so that the algorithms can behave intelligently as well. Such successful mappings include schools of fish, ant colonies and termites, foraging behaviour of bees etc. Karaboga [6] came up with the bee colony optimization algorithm in 2005. The algorithm has been proved to be very efficient on many applications than the earlier evolutionary algorithms, but has a little implementation on N-Queen problem. The problem now advances to

"Implement the artificial bee colony algorithm and modified differential evolution algorithm, compare the results of modified differential evolution results with that of classical differential evolution presented in [1], mark the improvement ,then compare the improved results with the results of artificial bee colony algorithm and note the drift towards optimization".

3.5 Objectives

i) The objective of the study is to modify differential evolution algorithm and apply on N-Queen problem to get faster convergence rate, than the earlier original algorithm. Success rate, Success performance and system time are also measured on each N size of up to 20.

ii) The swarm optimization algorithm artificial bee colony (ABC) is implemented on the N-Queen problem and the results are noted as Number of function calls, Success Rate, Success Performance and System time.

iii) The modified differential evolution algorithm results are evaluated against those of original differential evolution algorithm and against the results of artificial bee colony algorithm.

iv) Present the best algorithms and strategies to solve the combinatorial problems like N-Queen.

v) There is situation in selection method of differential evolution algorithms where not all the best individuals are selected, find an appropriate method to solve this issue and see if it works well with the differential evolution.

3.2 Work methodology

The working methodologies would be different for differential evolution algorithm and the artificial bee colony algorithm. The differential evolution would involve the steps of initialization of population followed by mutation, crossover and selection. The mutation step of the differential evolution algorithm is changed in this work and a new mutation strategy presented in [2] is used. The new mutation strategy comes with the advantage of non-use of the necessary repair function which had to be used in the conventional mutation strategy as it is used to remove the duplicate values from the candidate solution that exists due to the nature of mutation techniques used.

Step 1- Initialization of the population

Population in DE is initialized with assigning each candidate solution the initial values from zero to the N-1. Then each vector is shuffled with the no of times that is generated randomly. It ensures the randomness of the candidate solutions. Next comes the mutation step

Step 2- Mutation

The new mutation strategy defines the difference between the two different candidate solutions as list of movements and is as

Modified mutation proposed by Shahryar Rahnamayan et.al. [1]

Definition 1:- The application of valid movements m_k of differential list of movements $M_{p\to k}$ to solution $S_p \in S$ leads to a solution $S_k \in S$.

$$M_{p\to k} = S_p \ominus S_k \tag{3.1}$$

Where $\ominus$ returns a list of movements. The application of $M_{p\to k}$ to S_i is defined as:

$$s_i' = s_i \oplus M_{p\to k} \tag{3.2}$$

Where the operator $\oplus$ receives an applicable solution and a list of movements, returning another solution.

$$s_i = s_j \oplus M_{p\to k} = s_j \oplus (s_p \ominus s_k) \tag{3.3}$$

In this work, the multiplication of differential list of movements by a mutation factor F returns a list with first $|F*M_{p\to k}|$ movements of $M_{p\to k}$, where $|M_{p\to k}|$ is the size of list.

In this study the value of F is taken as 1. This means the application of every movement in the list to any solution vector. The classical original crossover proved to be less effective with the above mutation strategy. The new crossover technique is used that enhances the convergence rate of the differential evolution algorithm to greater extent.

Step 3- Crossover

New Crossover technique used in this work

The new crossover technique used in this worker has been proposed as one of the simplest standard crossover technique but has not found any implementation in differential evolution algorithm. In this work it was found that applying it with the conventional mutation didn't find any results but when the same crossover was used in conjunction with the above mutation strategy the results were exceptionally improved, so we propose the de with the conjunction of modified mutation and crossover strategy for solving combinatorial problems. The new crossover works as

Cr_Pt = rand (0, D)

Where D is the vector dimension.

If crossover_rate > rand (0, 1)

$$U_{i,j} = v_{i,j} \quad j=0\ldots Cr\_Pt \text{ and}$$

$$U_{i,j} = x_{i,j} \quad j=Cr\_Pt\ldots D$$

Else (3.4)

$$U_{i,j} = v_{i,j} \quad j=Cr\_Pt\ldots D \text{ and}$$

$$U_{i,j} = x_{i,j} \quad j=0\ldots Cr\_Pt$$

Where $U_{i,j}$ is the generated trial vector . *Cr_Pt* is the point of joining the initial vector and mutated vector in the trial vector.

Step 4- Selection

Selection is the process where out of initial vector and trial vector, a vector with best fitness value is chosen for the next generation. The flowchart of the different processes in differential evolution algorithm is below.

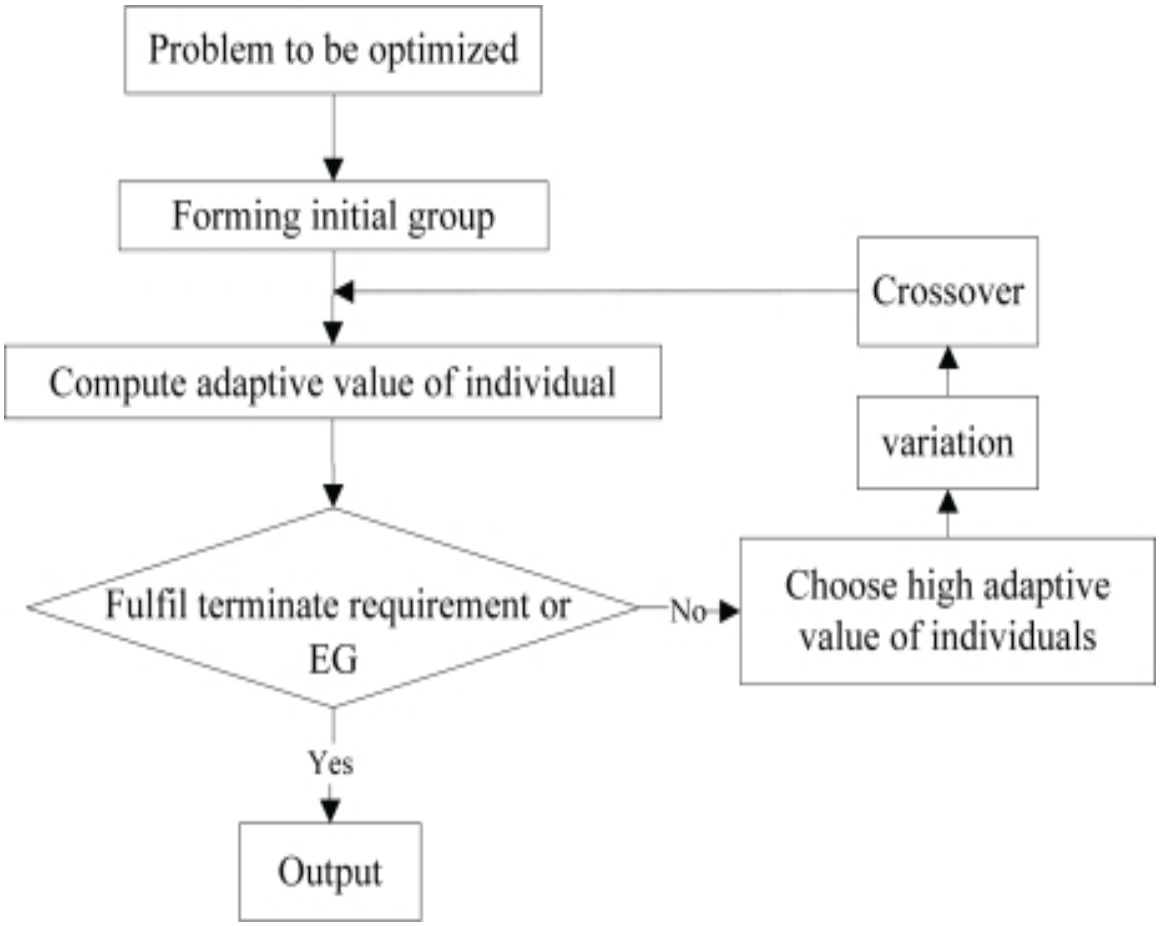


Figure 2: Differential Evolution Mechanism

EG denotes the end generation, Variation represents the mutation process and the adaptive value represents the fitness value of the candidate solutions. The methodology followed in artificial bee colony algorithm (ABC) is described as

Step 1- Initialization

Population in artificial bee colony algorithm is initialized with assigning each candidate solution the initial values from zero to the N-1. Then each vector is shuffled with the no of times that is generated randomly. It ensures the randomness of the candidate solutions.

Step 2- Send Employed bees to work

Employed bees are sent to optimize the solution. A random number is generated with population range that denotes the neighbour solution of the candidate solution under study. The neighbour solution is used to produce the mutant solution of the solution under process. The whole process is accomplished in the following manner

1. Getting the fitness value of the solution.
2. Randomly generate the number that determines the parameter to be changed.
3. Calculate the new value with the help of neighbour solution.
4. Trap the value within upper bound and lower bound limits.
5. Get the index of new value.
6. Swap the values.
7. Apply greedy selection to choose between the original solution and the modified solution. The selection is based on which of the two has the best fitness value.

Step 3- Calculate the fitness values of all the candidate solutions in the population.

Step 4- Calculate the probability values associated with all the solutions in the population.

Step 5- Send Onlooker bees

The onlookers are send to optimize the solution. The best solution from population is worked on by the onlookers. Here best solutions have the highest selection probability. Here again based on the random value the current bee and the neighbour bee are send to work that works same as that for employed bees.

Step 6- Memorize the best food source

The best of the solution is memorized as gBest.

Step 7- Send Scout bees

It finds the food sources which have been abandoned or reached the limit. Scout bees generate a totally random solution from the existing one and reset its trials back to zero.

The whole process from the step 2 to step 7 is iterated till the solution is met or the maximum limit of epochs is reached.

3.3 Tool and Language Used

The tool used for coding in this research work is the IBM developed eclipse. The language used is java.

3.4 Scope of the study

The task of finding the best algorithm for solving the combinatorial problems will lead to the optimization in the following practical applications.

The N-queen solution in particular will be helpful in

i) Parallel storage schemes.

ii) VLSI testing.

iii) Traffic control and deadlock prevention etc.

iv) Data/message routing.

v) Load balancing in multiprocessor computers.

vi) Data compression.

vii) Computer task scheduling.

viii) Optical parallel processing.

ix) Air traffic control.

x) Modern communication systems.

Finding the best algorithm will also lead to the following application optimization.

xi) This will also lead to the development of best airline network of spokes and destinations.

xii) Determination of the optimal route to deliver packages etc.

xiii) Construction of railways, roads, etc.

xiv) The determination of the right attributes of concept elements before concept testing.

CHAPTER 4

RESULTS AND DISCUSSION

The metrics used for comparison are number of function calls (NFC), success rate (SR), and success performance. The number of function calls (epochs) measures the convergence speed [1]. The lower value of NFC implies higher convergence speed and vice versa. The termination criteria is to reach a solution where there are mutually non attacking queens i.e. the total conflicts of the board equals to zero. It's defined as the value to reach (VTR). The success rate is defined as:

$$\text{SR} = \frac{number\ of\ times\ VTR\ reached}{total\ number\ of\ runs} \tag{4.1}$$

The NFC and SR are the main metrics in this optimization process. Combining the two in a single metrics defines the Success performance (SP).

$$\text{SP} = \frac{MEAN\ (NFC\ for\ successful\ runs)}{SR} \tag{4.2}$$

The average time taken by the total runs is also measured to compare the two algorithms. The lesser is the average time taken on any N-value, faster the convergence. The average success rate is calculated as:

$$SR_{avg} = \frac{1}{n}\sum_{i}^{n} SR_i$$

The results of application of DE to solve N-Queen problem (N=4 to N=20) are in Table I.

Table. I

Comparison of ABC and DE. N by N chess board. Dimensions, NFC: Number of function calls. The number of trials =20, SR: Success Rate, SP: Success Performance.

Table 1: The results of application of DE on N-Queen problem (N=4 to N=20)

	PROPOSED DIFFERENTIAL-EVOLUTION			
N	NFC	SR	SP	AVG.TIME_FOR_SUCEESFUL_RUNS_(ns)

4	1	1.0	1	1379199
5	1	1.0	1	1614157
6	5	1.0	5	3560211
7	3	1.0	3	3860155
8	14	1.0	14	9317533
9	35	1.0	35	24003825
10	209	1.0	209	161960709
11	764	1.0	764	692540874
12	1620	1.0	1620	1719358739
13	2521	1.0	2521	2925978158
14	11533	1.0	11533	15613531759
15	15316	0.75	20421.33	17383345586
16	16892	0.65	25987.69	18786160626
17	19856	0.40	49640	14847620844
18	18499	0.10	184990	4018037259
19	26522	0.15	176813.33	9143995052
20	-	-	-	-
AVG_SR		0.83		
∑			7287558.4	

The line graph next shows a trend on NFC against N values. As the value of N goes on increasing the number of function calls to arrive at a solution also grows. Up to size of ten the number of function calls (NFC) are close to each other but beyond that the graph shows certain variations. The NFC values are on high between the intervals of 13 to 15. At N size of 17 the graph shows a sudden steepness, it means that at the N value of 18 the number of function calls required to converge to a solution are less than what was at 17. Beyond this there is again a rise in the line showing further increase in NFC values with N.

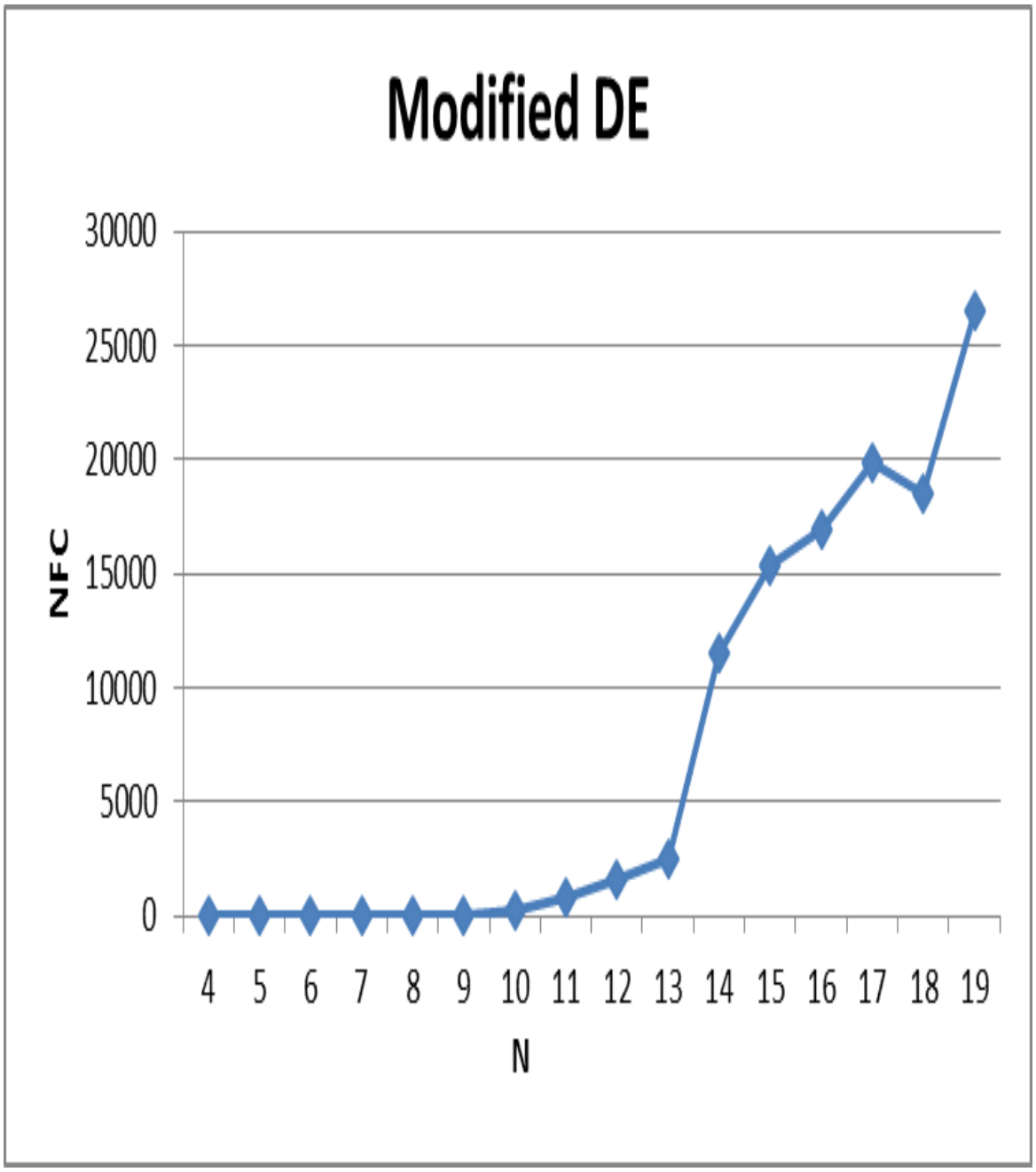


Figure 3: Number of Function Calls plotted against N size for modified-DE

The below line graph shows success rate against N values for the proposed DE.

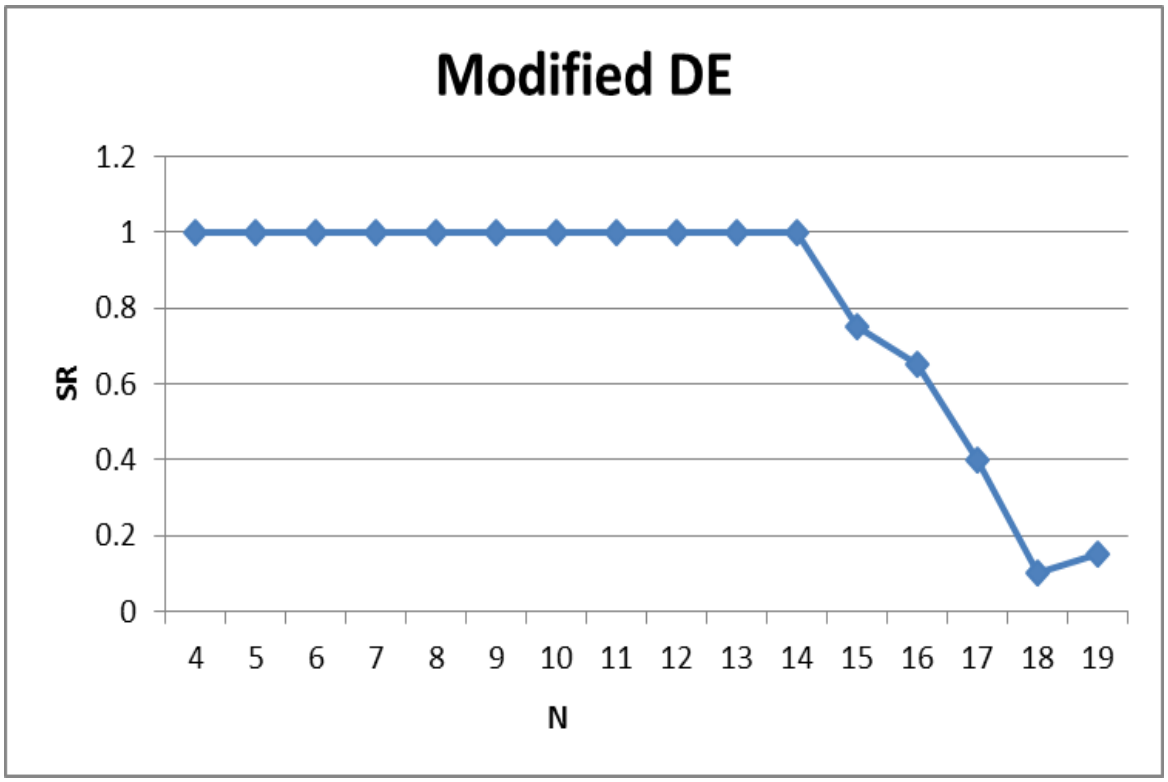


Figure 4: Success Rate plotted against N size for modified-DE

Parameter settings for all the experiments conducted in this research are as follows:

- Common settings for DE and ABC
  - Population Size = 40
  - Value to reach, VTR=0 (mutually non-attacking queens)
  - Max Function calls = 40000
  - Trial limit / MAX RUNS = 20
- Settings for DE
  - Mutation factor=1
  - Crossover probability=0.9
  - Mutation strategy: Differential List of movements
- Settings for ABC
  - Food Number = Population size/2
  - Selection probability factor = 0.9

With these settings the modified differential evolution produced the results noted down in Table I. The results when compared against those produced in [1] showed exponential

growth in performance. The success rate proved to be very high. The results produced by Shahryar Rahnamayan et.al. [1] are below:

Table 2: The results of application of Classical DE on N-Queen problem (N=4 to N=15)

	CLASSSICAL DIFFERENTIAL EVOLUTION [1]		
N	NFC	SR	SP
4	134	1.00	134
5	254	1.00	254
6	111136	0.65	170979
7	24338	0.95	25619
8	7576	0.75	10101
9	19296	0.50	38592
10	286208	0.30	954028
11	68255	0.10	682550
12	99120	0.25	396480
13	95485	0.15	636570
14	160475	0.10	1604750
15	223425	0.10	2234250
AVG_SR		0.49	
∑			6754306

The experiments with the classical DE clearly show that at N value of 15 the only two runs of the algorithm converge to perfect solution and also it needs about two lakh twenty three thousand and four hundred twenty five function calls. The same value of

N in case of modified DE takes only 15,316 function calls which shows huge reduction in the number of function calls and the enhancement to the convergence rate. N value of 15 has about 15 runs out of 20 which converge to solutions against that of 2 out of 20 in case of classical DE. The average success rate in case of classical DE is about 0.49 which is 0.83 in case of modified DE with 4 to 19 values of N against 4 to 15 values of N in DE used in [1]. The average success rate in case of proposed DE for N values from 4 to 15 is 0.977 against 0.49 of classical DE. The proposed DE used in this research with the set parameters starts to fall in success rate as size of N increases from 15. The graphical plotting of different values of N against NFC is shown below in fig. 3 and it indicates the variations in number of function calls for each N value between original DE and the proposed DE.

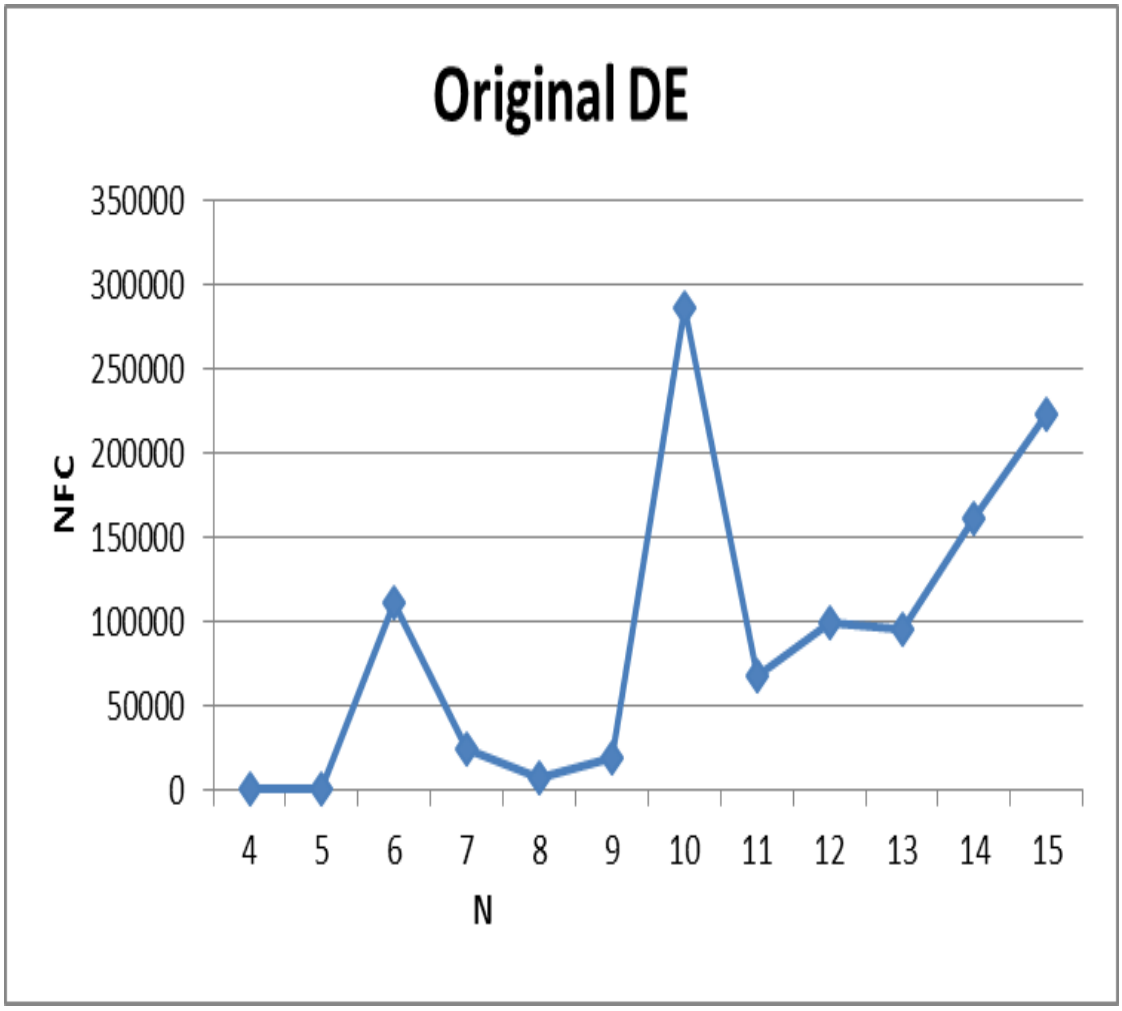


Figure 5: Number of Function calls plotted against N size for original-DE

The next graphical plot shows success rate against N value for original-DE.

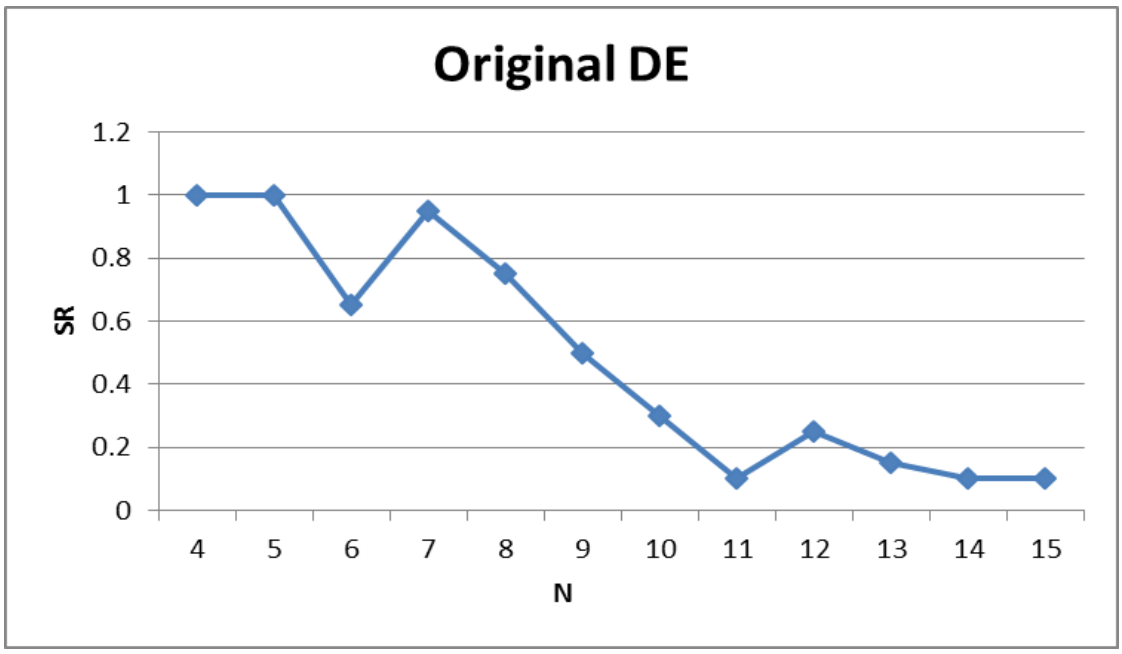


Figure 6: Success Rate plotted against N size for original-DE

The comparative line graphs below show direct performance difference between the original classic DE and modified DE.

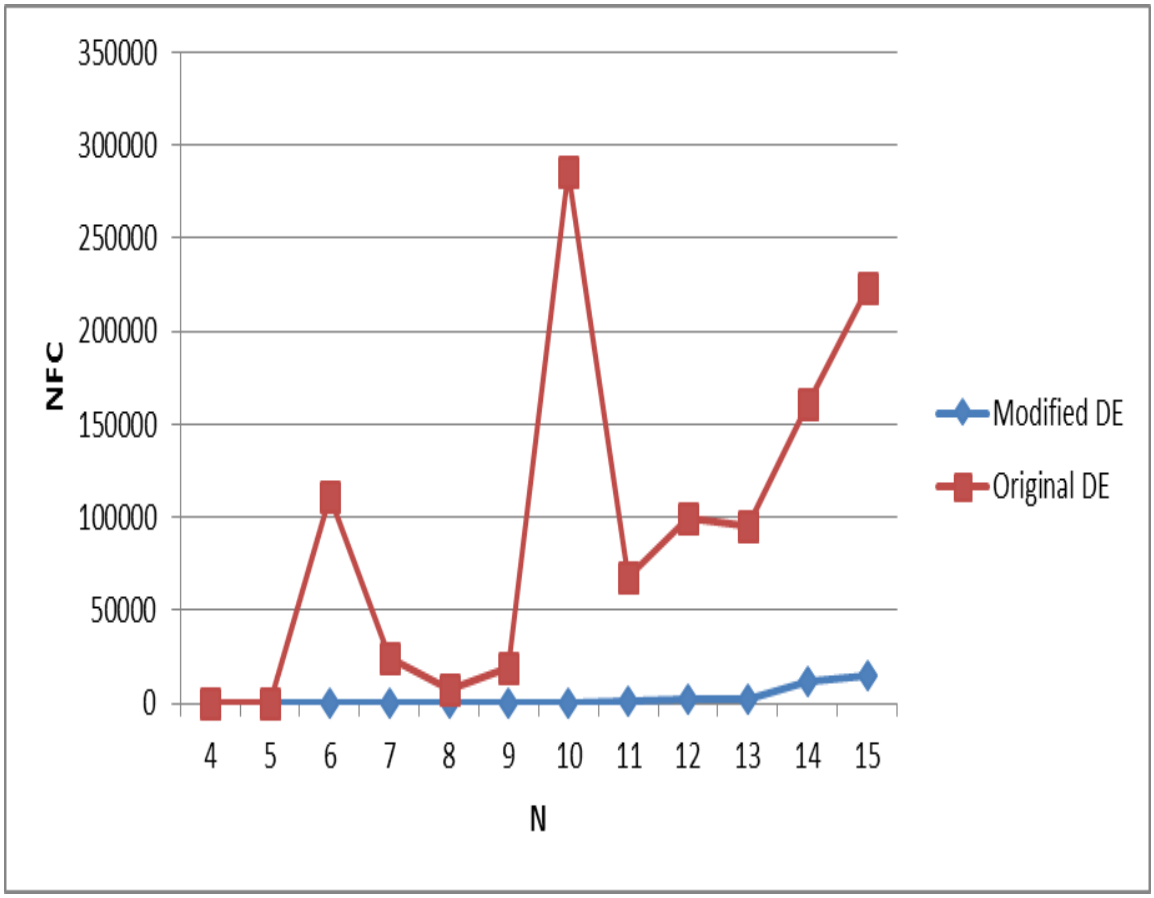


Figure 7: NFC plotted against N size for original DE and modified DE

The line graph below clearly depicting the strike rate comparisons between original DE and modified DE.

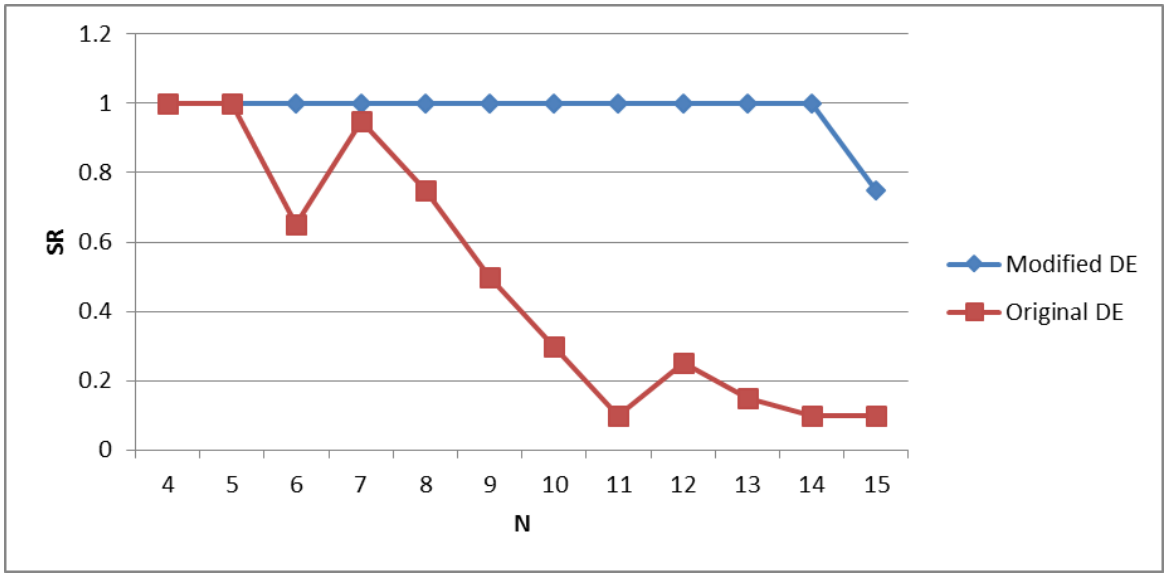


Figure 8: Success Rate calls plotted against N size for modified DE and original DE

The application of ABC algorithm on N-Queen problem generated the following results.

Table 3: The results of application of ABC on N-Queen problem (N=4 to N=20)

	ARTIFICIAL BEE COLONY ALGORITHM			
N	NFC	SR	SP	AVG.TIME_FOR_SUCEESFUL_RUNS_(ns)
4	1	1.0	1	1194723
5	1	1.0	1	1326017
6	15	1.0	15	3839060
7	4	1.0	4	2204024
8	7	1.0	7	3176294
9	10	1.0	10	4322481
10	37	1.0	37	11727561

11	38	1.0	38	1344842
12	61	1.0	61	22562989
13	79	1.0	79	32858006
14	79	1.0	79	36462082
15	144	1.0	144	70053700
16	142	1.0	142	78653098
17	364	1.0	364	221743459
18	400	1.0	400	272912630
19	804	1.0	804	2598866021
20	1258	1.0	1258	972588540
AVG_SR		1.00		
∑			3444	

The results of ABC completely outperform the original DE and proposed DE as well. It may be due to its swarm intelligent approach for solving the problems. The graphical plot of ABC on NFC value against N value clearly shows the performance enhancement. The graphical plots below shows ABC results of NFC and SR against N values along with the original DE. It gives a clear indication of the performance comparison. The line graph in the Figure 4(h) shows variation of success rate with increasing size of N of ABC and DE. From 0 to 15 N size the success rate of ABC remains uniform and maximum. It means that all the runs converge to solutions which is not the case with differential evolution algorithm. The DE perfectly shows a maximum success rate only on the first two values, 4 and 5. Success rate of DE continuously falls with every increase in N-value except at 7 and 12 where surprisingly the success rate shows a sudden jump. At value of 15 the success rate of DE is down to 0.10 while ABC still performs optimally. Comparison between ABC and modified DE showed much

enhance in performance of DE and shows greater convergence towards the optimal solution.

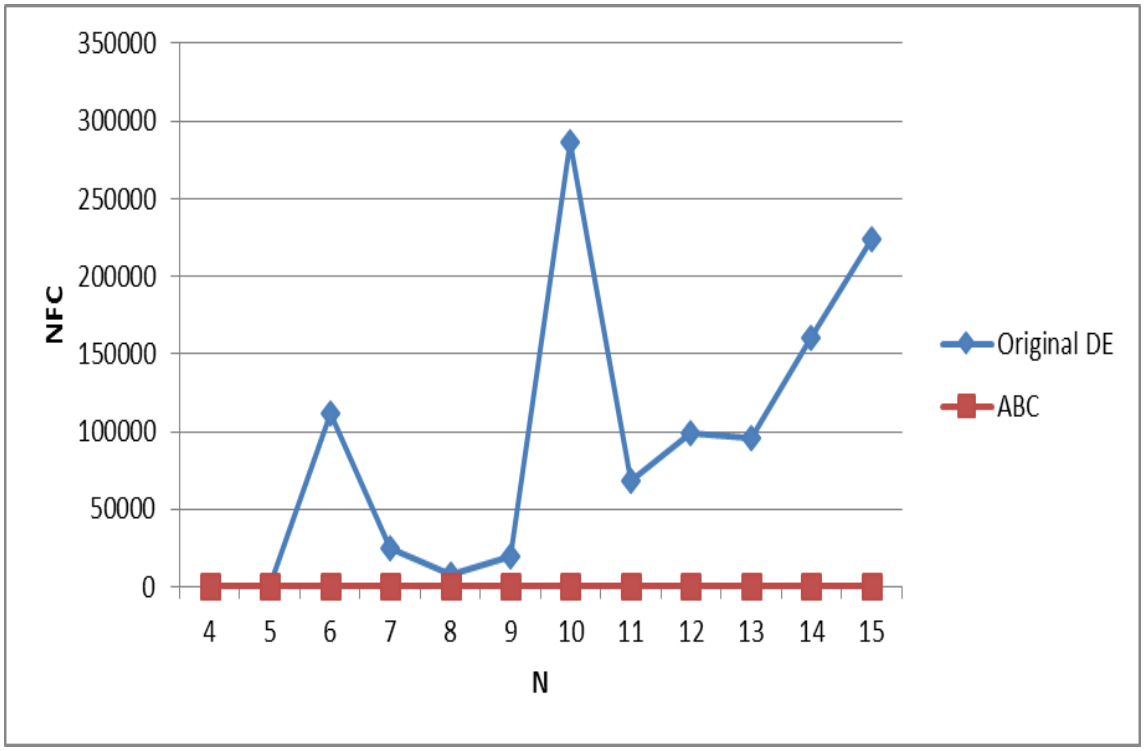


Figure 9: Number of Function calls plotted against N size for original-DE and ABC

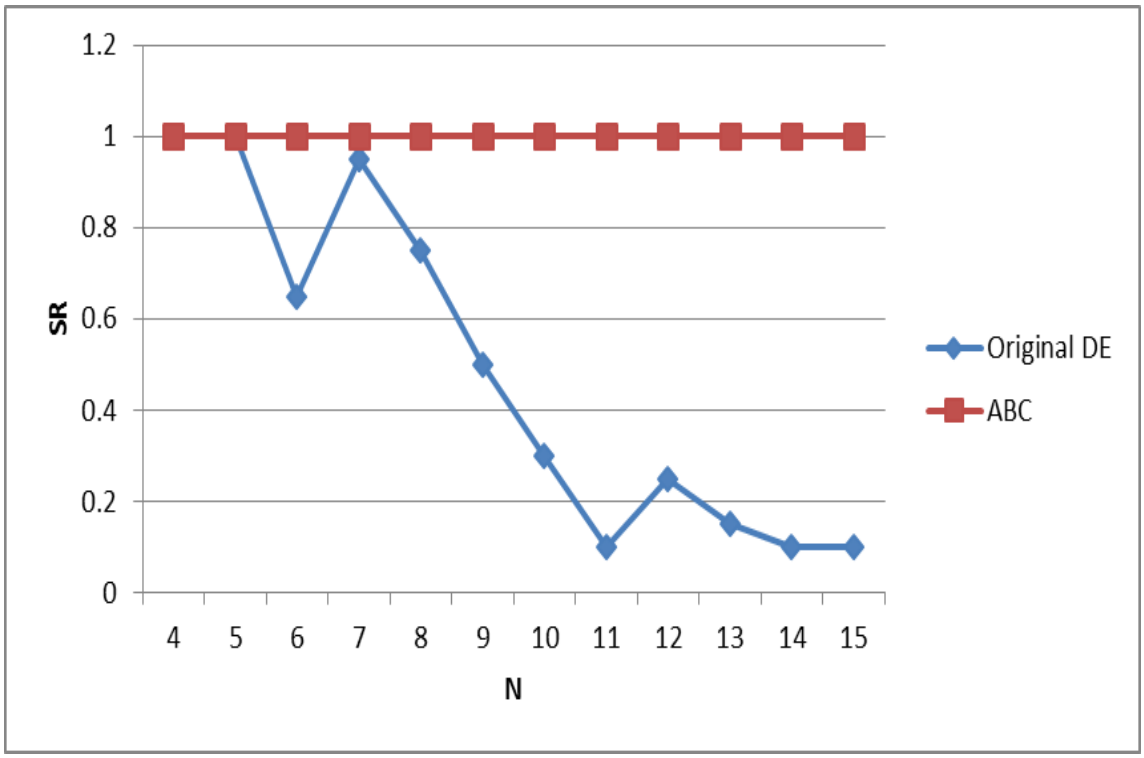


Figure 10: Success Rate plotted against N size for original-DE and ABC

Below line graphs next shows comparison of ABC with modified DE.

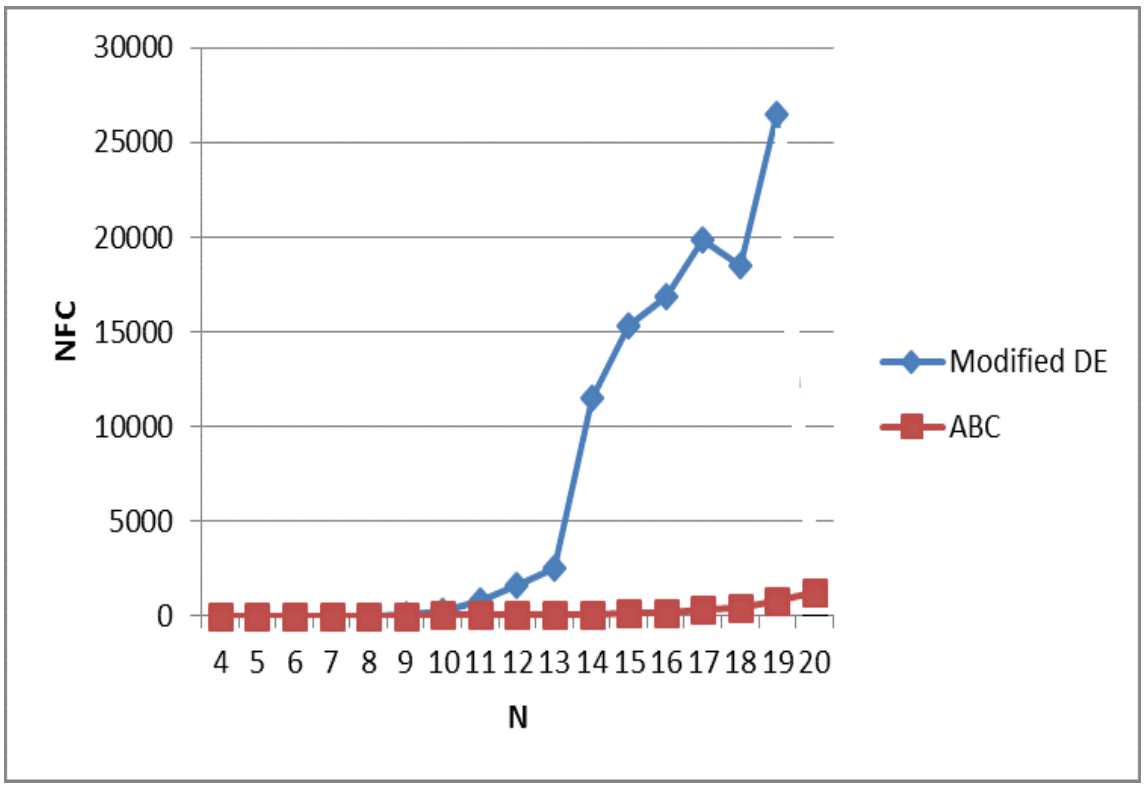


Figure 11: Number of Function calls plotted against N size for modified-DE and ABC

The above line graph shows comparison between number of function calls needed for convergence between artificial bee colony algorithm and modified differential evolution.

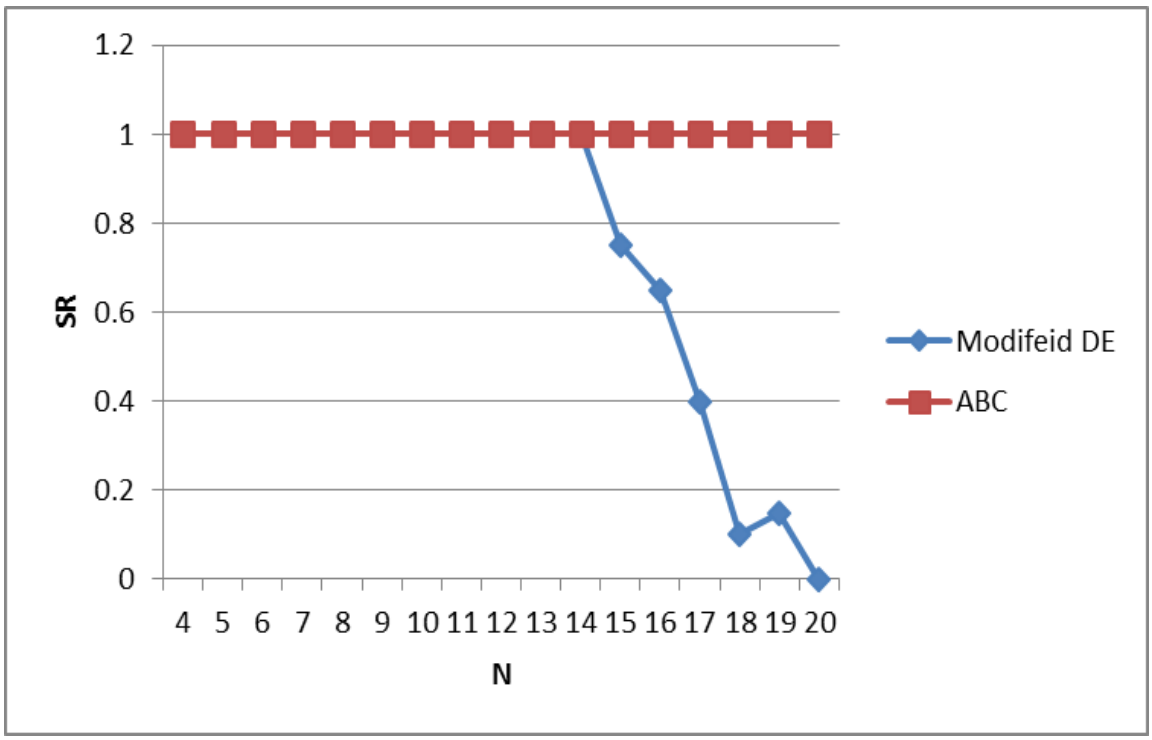


Figure 12: Success Rate plotted against N size for modified-DE and ABC

The above line graph shows comparison between success rates with different N sizes in case of artificial bee colony algorithm and modified differential evolution. Sample cuts from the solution generated by implemented code.

```
Epoch: 4
done.
Completed 4 epochs.
SOLUTION
Board:
. . . . . . . . . Q
. . Q . . . . . . .
Q . . . . . . . . .
. . . Q . . . . . .
. . . . . . Q . . .
. . . . . . . . Q .
. Q . . . . . . . .
. . . . Q . . . . .
. . . . . . . Q . .
. . . . . Q . . . .
conflicts:0
Done
run 20
time in nanoseconds: 1428816
Success!
Number of Success: 20
Number of failures: 0
Average Success time :8946846
Average number of function calls required 20
run with time 12503356 nanoseconds
run with time 19148296 nanoseconds
run with time 21752576 nanoseconds
run with time 15718057 nanoseconds
run with time 1801010 nanoseconds
run with time 6308398 nanoseconds
run with time 4706720 nanoseconds
run with time 5468935 nanoseconds
```

Figure 13: Screenshot of result, 10 by 10 chessboard (ABC algorithm)

```
Epoch: 79
Epoch: 80
done.
Completed 80 epochs.
SOLUTION
Board:
. . . . Q . . . . .
. . . . . . . Q . .
. Q . . . . . . . .
. . . . . . Q . . .
. . . . . . . . . Q
. . Q . . . . . . .
Q . . . . . . . . .
. . . . . . . . Q .
. . . Q . . . . . .
. . . . . Q . . . .
conflicts:0
Done
run 20
time in nanoseconds: 59639700
Success!
Number of Success: 20
Number of failures: 0
Average Success time :172396921
Average number of function calls required :
Success Rate = 1.0
Success Performance 224.0
run with time 30929951 nanoseconds
run with time 299842857 nanoseconds
```

Figure 14: Screenshot of result, 10 by 10 chessboard (modified-DE algorithm)

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

The purpose of this research is not to propose a novel and better heuristic technique for N-queen combinatorial problem, Instead we are using this problem to compare different algorithms for using in combinatorial optimization. The algorithms we are working on are differential evolution algorithm and artificial bee colony algorithm. Different parameters will be used for evaluation that includes number of function calls (NFCs), success rate (SR) and success performance (SP). The convergence of both algorithms will be measured on different inputs given by NFC value. The research will propose an efficient and fast algorithm for solving many combinatorial problems.

This research concludes on proposing a better and efficient differential evolution algorithm that solves combinatorial problem, N-Queen in our case, efficiently that the original classic differential evolution algorithm, proposed by Karaboga. Also the results from the modified differential evolution and ABC clearly shows that the later algorithm outperforms the proposed DE in all the cases. Since DE finds immense application in computing world, the proposed research may be helpful to optimize the implementations.

5.1 Future Scope

The future work of this research will try to analyse the selection method of the differential evolution algorithm and if there is any scope to converge towards the optimality that we obtained in case of artificial bee colony algorithm. This is because on observing the selection function, I noticed following

Selection method is a process in which between a trial candidate solution (generated by crossover operator) and target (initial) solution, solution with best fitness value is chosen. But this comparison is one to one comparison in which first trial solution is compared with the first of initial target solution. This comparison makes sure the final population post selection process is of better fitness valued candidates but does not guarantee it to be the best as this selection process sometimes is responsible for omitting the best fitness candidates.

CHAPTER 6

REFERENCES

[1] Rahnamayan, S., Dieras, P., "Efficiency competition on N-Queen problem: DE vs. CMA-ES", *IEEE Electrical and Computer Engineering, CCECE, Canada*, pp.33-36, 2008.

[2] Prado, R.S., Silva, R.C.P., Guimaraes, F.G., Neto, O.M., "Using Differential Evolution for Combinatorial Optimization". *IEEE Systems Man and Cybernetics (SMC), International Conference*, pp.11-18, 2010.

[3] Hegerty, B., Hung, C.C., Kaspark, K., "A comparative study on differential evolution and genetic algorithms for some combinatorial problems".

[4] Abro, A.G., Saleh, J.M., "Intelligent Scout-Bee based artificial bee colony optimization algorithm", *IEEE International Conference on Control system, Computing and Engineering, Penang, Malaysia*, pp.380-385, 2012.

[5] Wong, L.P., Low, M.Y.H., Choong, C.S., "A bee colony optimization algorithm for travelling salesman problem", *IEEE Second Asia International Conference on Modelling & Simulation*, pp.818-823, 2010.

[6] Karaboga, D., Akay, B., "A Comparative study of artificial bee colony algorithm. Applied Mathematics and Computation", *Elesiver*, pp. 108-132, 2009.

[7] Storn R., and Price K., "Differential Evolution – A simple and efficient adaptive scheme for global optimization over continuous spaces", *Journal of Global Optimization, Kulwer Academic Publishers*, pp.341-359, 1997.

[8] Karaboga, D., (2005). An Idea based on honey bee swarm for numerical optimization, TECHNICAL REPORT-TR06, Erciyes University.

[9] Karaboga, D., Ozturk, C., A Novel Clustering approach: Artificial Bee Colony (ABC) algorithm, *Applied Soft Computing ,ELESIVER*, vol. 11pp.652-657, 2011.

[10] Davis, L., Ed. Genetic Algorithms and Simulated Annealing, Pitman, London, 1987.

[11] Dorigo, M., Optimization, Learning and Natural Algorithms, PhD thesis, Politecnico di Milano, 1992.

[12] Kennedy, J., Eberhart, R., "Particle Swarm Optimization", *IEEE Neural Networks Proceedings, Proceedings, International Conference,* pp.942-948, 1992.

[13] Bozikovic, M., Golub, M., Budin L., "Solving N-Queen problem using parallel genetic algorithm", *EUROCON.Computer as a tool. The IEEE Region 8*, vol (2), pp.104-107.

[14] Sinha, S.N., Palit, S., Molla, M.A and Khanra, A., "A cryptanalytic attack on knapsack cipher using differential evolution algorithm", *IEEE Recent Advances in Intelligent Computational Systems (RAICS),* pp. 317-320, 2011.

[15] Turky, A.M and Ahmad, A., "Using genetic algorithm for solving N-Queens problem", *IEEE Information Technology (ITSim), International Symposim,* vol. 2, pp.745-747, 2010.

[16] Turner, J., Ali, S., "The N-Queens problem and genetic algorithms", *IEEE Southeastcon '92, Proceedings,* vol. 1, pp.262-267, 1992.

[17] Rok sosic and Jun Gu., "Efficient local search with conflict minimization: a case study of n-queens problem", *IEEE Knowledge and data engineering,* vol. 6, pp.661-668, 2002.

[18] Heris, J.E.A and Oskoei, M.A., "Modified genetic algorithm for solving N-Queens problem", *IEEE Intelligent systems (ICIS), Iranian conference,* pp. 1-5, 2014.

[19] Xiao-Bing Hu., Leeson, M.S and Hines, E.L., "A review on ripple-spreading genetic algorithms for combinatorial optimization problem", *Cognitive Informatics (ICCI), 9th IEEE International Conference,Beijing*, pp. 441-448, 2010.

[20] Yu Liang., Yu Liu and Liang Zhang., " An improved artificial bee colony (ABC) algorithm for large scale optimization", *Instrumentation and Measurement, Sensor Network and Automation (IMSNA),2nd International Symposium, Toronto,* pp. 644-648, 2013.

[21] Abu-Mouti and F.S., El-Harway., "Overview of artificial bee colony (ABC) algorithm and its applications*", Systems Conference (SysCon), 2012 IEEE International, Vancouver, BC,* pp.1-6, 2012.

[22] Babayigit, B and Ozdemir, R.., "An ABC algorithm with inversely proportional mutation", *Innovations in Intelligent Systems and Applications (INISTA), International Symposium, Trabzon,* pp.1-5, 2012.

[23] Martinjak, I and Golub, M., "Comparison of heuristics for N-Queen problem", *Information Technology Interfaces, ITI, 29th International Conference, Caytat*, pp.759-764, 2007.

[24] Shima Sabet., Fardad Farokhi and Mohammad Shokouhifar., "A hybrid mutation-based artificial bee colony for travelling salesman problem", *Lecture notes on information theory*, vol.1, pp.99-103, 2013.

[25] Musarat Ali., Millie Pant and Ajith Abraham., "Simplex differential evolution", *Acta Polytechnica Hungaria*, vol. 6, pp.99-115, 2009.

[26] Dervis Karaboga and Bahriye Basturk., "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm", *Springer science+business Media B.V*, pp.460-471, 2007.

[27] Ming-Huwi Horng., "Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation", M.-H. Horng / Expert Systems with Applications 38, pp. 13785-13791, 2011.

CHAPTER 7

APPENDIX

ABBREVIATIONS

a) DE - Differential Evolution
b) ABC - Artificial Bee Colony
c) NFC - Number of Function Calls
d) SR - Success Rate
e) SP - Success Performance
f) EA - Evolutionary Algorithm
g) GA - Genetic Algorithm
h) PSO - Particle Swarm Optimization
i) TSP - Travelling Salesman Problem