OPTIMIZATION OF A MULTI-DIMENSIONAL PROBLEM BY AN ARTIFICIAL BEE COLONY ALGORITHM WITH A NEIGHBOURHOOD MUTATION OPERATOR

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

OF

MASTER OF ENGINEERING IN
ELECTRONICS & TELE-COMMUNICATION ENGINEERING
IN SPECIALIZATION OF CONTROL ENGINEERING 2017-19

By:

ARNAB KUMAR SEN

ROLL NO.: 001710702025

UNIVERSITY REGISTRATION NO.: 140708 OF 2017-18

EXAMINATION ROLL NO.: M4ETC19025

Under the supervision of

DR. PRATYUSHA RAKSHIT

Assistant Professor

DEPARTMENT OF ELECTRONICS & TELE-COMMUNICATION ENGINEERING

JADAVPUR UNIVERSITY

KOLKATA-700032

MAY 2019

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

JADAVPUR UNIVERSITY

CERTIFICATE OF RECOMMENDATION

This is to certify that the dissertation entitled **"OPTIMIZATION OF A MULTI-DIMENSIONAL PROBLEM BY AN ARTIFICIAL BEE COLONY ALGORITHM WITH A NEIGHBOURHOOD MUTATION OPERATOR"** submitted by **ARNAB KUMAR SEN (Examination Roll No.: M4ETC19025, University Registration No.: 140708 of 2017-18)** to Jadavpur University, Kolkata, is a record of bonafide research work under my supervision and be accepted in partial fulfilment of the requirement for the degree of **Master of Engineering in Electronics and Telecommunication Engineering** of the institute. The research results presented in this thesis are not included in any other paper submitted for the award of any Degree or Diploma to any other University or Institute. The project in my opinion, is worthy for its acceptance.

DR. PRATUSHYA RAKSHIT

Thesis supervisor
(Assistant Professor)
Department of Electronics and Telecommunication Engineering
Jadavpur University
Kolkata 700032

PROF. SHELI SINHA CHAUDHURI
Head of the Department
Department of Electronics &
Telecommunication Engineering

PROF. CHIRANJIB BHATTACHARJEE
Dean
Faculty of Engineering and Technology
Jadavpur University

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

JADAVPUR UNIVERSITY

CERTIFICATE OF APPROVAL

The foregoing THESIS is hereby approved as a creditable study of an Engineering Subject carried out and presented in a manner of satisfactory to warrant its acceptance as a pre-requisite to the degree for which it has been submitted. It is to be understood that by this approval, the undersigned do not necessarily endorse or approve any statement made opinion expressed and conclusion drawn therein but approve the THESIS only for the purpose for which it has been submitted.

Committee on Final Examination for The Evaluation of Thesis

Signature of the examiner	Signature of the supervisor
______________________	________________________

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I the undersigned do hereby declare that this thesis contains literature survey and original work done by means a part of my MASTER OF ENGINEERING IN ELECTRONICS AND TELECOMMUNICATION ENGINEERING. All informations in this document have been obtained and presented in accordance with academic rules and ethical conduct. I also declare that as required by these rules and conduct I have fully cited and referenced all materials and results that are not original with this work.

Project Title

"OPTIMIZATION OF A MULTI-DIMENSIONAL PROBLEM BY AN ARTIFICIAL BEE COLONY ALGORITHM WITH A NEIGHBOURHOOD MUTATION OPERATOR"

ARNAB KUMAR SEN

Exam Roll No.: M4ETC

Registration No.: 140708 OF 2017-18

Electronics & Telecommunication Engineering Department,

Jadavpur University, Kolkata-700032, India.

May, 2019

Jadavpur University, Kolkata

(ARNAB KUMAR SEN)

Acknowledgement

The success and final outcome of my thesis required a lot of guidance, assistance and patience of my supervisor **Dr Pratyusha Rakshit** and I am extremely fortunate to have got this all along the completion of my thesis work. Whatever I have done has been possible only due to her guidance and assistance and I would like , to express my immense gratitude to her. Her invaluable help, guidance and criticism were the cornerstone of my dissertation work. I am highly indebted to her and I attribute my master degree to her encouragements and directions. No words will suffice in describing her contribution towards the completion of my thesis work.

I would also like to thank Prof. Sheli Sinha Chaudhuri, Head of the Department of Electronic & Telecommunication Engineering, Jadavpur University and all the authorities of the institute for providing nice academic environment and adequate infrastructure to carry out the present investigations.

Finally I would like to thank my parents, whose ever present love and guidance has been my primary motivation to do anything in life.

May, 2019

Jadavpur University, Kolkata

(ARNAB KUMAR SEN)

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 OPTIMIZATION

Optimization is the selection of the best possible solution from a given set of provided alternative with respect to a given set of solutions.[1]

In a simplistic explanation, a optimization problem consists of either maximizing or minimizing a real world problem by virtue of choosing real values from a given set of allowed values, and computing the given function through a set of performance evaluating functions otherwise known as *Benchmark Functions*. [2]

An indispensable part of the various important fields such as Mathematics and Computer Science, Optimization is at the heart of solving various practical problems. Many practical applications of Optimization include problems related to minimal cost , maximal profit, minimal error, optimal design, optimal management[3] and so on. Optimization is also widely employed in almost all natural processes, a brilliant example of which is *Natural Evolution.*[4] In the Darwinian evolution concept through the passage of millions of years every species on earth, in order to survive, has to adapt themselves to fit in their places in their respective environments, that are subjective to changes with time.

A basic observation of such natural phenomena can teach us to solve very complex problems both resourcefully and successfully. It so appears that nature has found a way to solve immensely difficult information processing algorithms. A keen observation of the underlying relation between optimization and complex natural processes like biological evolution led researchers to come up with a new and radical field known as *Nature Inspired Algorithms* (NIAs).[5] NIAs take their inspiration from some natural phenomena and apply it solve complex computational problems. Most important types of NIAs are :- The Evolutionary Algorithms, Physical Algorithms, Immune Algorithms, Neural Algorithms, Probalistic Algorithms, Stochastic Algorithms and the Swarm algorithms.

A basic example for understanding the concept of optimization would be by considering the example of a simple optimization problem as in :-

$$f(x) \ : A \dashrightarrow \mathbb{R} \qquad \dots (1.1)$$

That is let f(x) be a function a set A which comprises of real numbers and we may seek a value x_0 such that :-

$$\mathbf{x_0} \ \in \mathbf{A} \mid \mathbf{f(x_0)} \leq \mathbf{f(x)} \ \textbf{for all x} \ \in \mathbf{A} \qquad \dots (1.2)$$

or a value of x_0 such that :-

$$\mathbf{x_0} \ \in \mathbf{A} \mid \mathbf{f(x_0)} \geq \ \mathbf{f(x)} \ \textbf{for all x} \ \in \mathbf{A} \qquad \dots (1.3)$$

Equation (1.2) represents a ‘minimization problem’ whereas equation (1.3) represents a ‘maximization problem’. Such a formulation is known as the Optimization problem.

Typically ‘A’ is selected from a Euclidean space $\mathbb{R}^n$, and it is often specified by by a set of constraints, equalities or inequalities that the members of ‘A’ have to satisfy. The domain of ‘A’ is known as the ‘search space’ and the respective elements of ‘A’ are known as ‘Candidate solutions’ or ‘feasible solutions’. The function ‘f’ is referred to as objective function, loss function or cost function for minimization problems whereas for problems of maximization it is known as fitness function or utility function. The optimal solution of any optimization problem is the one that effectively minimizes or maximizes the given problem. Generally minimization problems are preferred as they are more convenient .

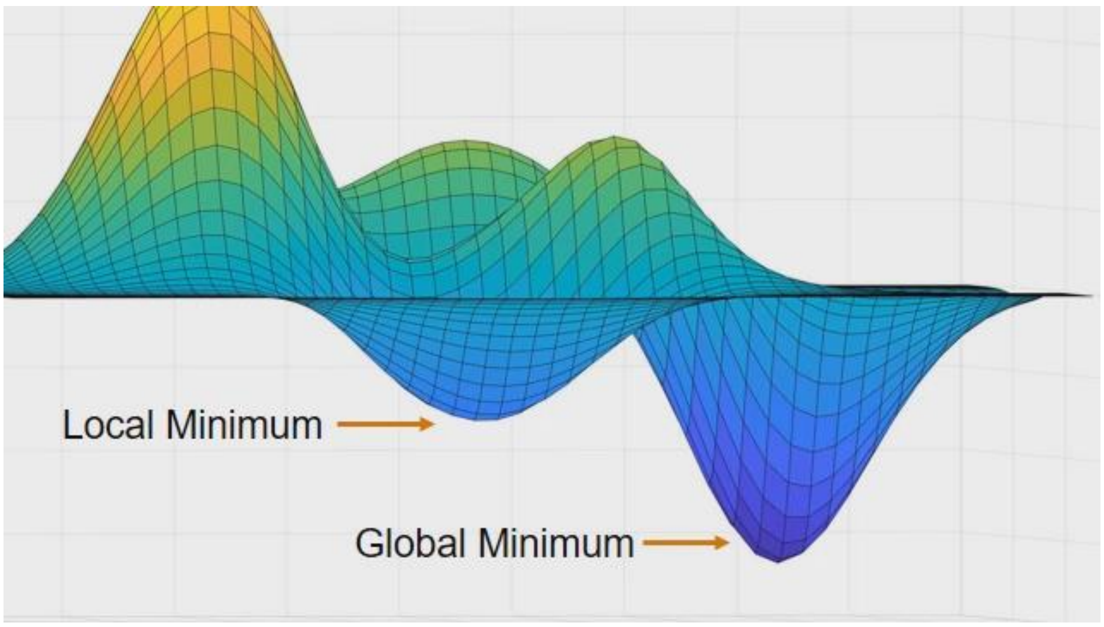


Figure 1.1 :- The various points of local and global minimums are shown in the figure.

Minimization problems can be resolved by using local minima as well as global minimums. Generally local minimas are sufficient for resolution of simple problems, however if the given function is a convex problem then there is a problem of several local minimas occurring simultaneously over several points in the problem function. An example of a convex function would be the spherical function with multiple local minima, while having have a single global minima.

1.1.1 Multi Objective Optimization

Multi-Objective Optimization [6] or Multi-attribute optimization or Pareto Optimization is specialized form that takes on the task of optimizing multiple objectives simultaneously. Muti-Objective Optimizations are particularly useful where the required task at hand, has one or more conflicting objectives. For example say there is the task of building a cantilever bridge over a river. The bridge must be strong and stable enough to withstand the load passing over it, while also remaining flexible enough to counter the wind shear. Thus there are several conflicting parameters that have to be taken into account. There would a strongest and stiffest bridge, on the opposite end there must also be a most flexible design. At the same time there may be infinitesimally different numbers of designs having some sort of a trade-off between strength and flexibility. This is where Multi-Objective Optimization [7] comes into the picture. The actual bridge that would be constructed would be done after deliberating through the design with the best possible trade-offs between strength and flexibility. This is done by examining the possibilities of the various trade-offs by examination of the Pareto set. The Pareto set [8] can be defined as the set of the various choices of trade-off designs such that maximizing one trade-off will definitely result in a negative impact upon a other trade-off design. The Pareto sets can be graphically represented as a Pareto Frontier where two conflicting trade-off designs are plotted against each other. A design is said to be Pareto Optimal where there is not a clear favouritism to one of the conflicting trade-offs, and there is an optimal balance between the two conflicting design objectives. Practical designs have several Pareto Optimal solution specified out of which one is implemented by the engineer or designer.

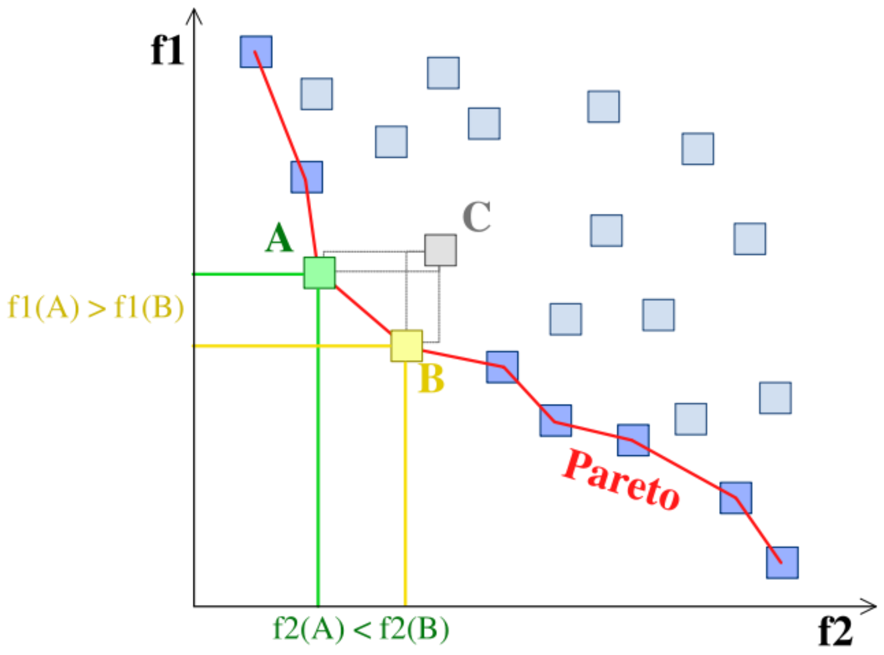


Figure 1.2 :- A Pareto frontier is shown. The set of pareto optimal solutions marked by boxes not dominated by any other solutions has been marked in red Solution C is dominated by solutions A & B hence is excluded from the Pareto frontier.

Multi-objective optimization is widely used in several diverse fields like industrial manufacturing as well as in Economics. An example of would be the design of a budget segment car. The manufacturer must balance the idea of providing the best possible comforts to the customer while cutting costs wherever possible. This would be drastic contrast to say the design of a high performance car where the objectives would to build a car that would be giving out the best performance possible while minimizing fuel consumption and limiting the exhaust of toxic pollutants to a bare minimum.

Multi-Objective Optimization also plays a crucial role in the fields like Industrial engineering and economics where the best alternatives can not always be clear cut. In such process the alternative trade offs may not always be definable as being the case the case of being ‘the better the best’ or ‘the least the better’. Rather there would be the case of desirability of getting the ideal values of every parameters rather than going for maximizing or minimizing a given alternatives. In such cases the most realistic approach would be achieve the ideal value of such interim objectives, and striving to get towards that desired values of every intermediate variables. An example would be the process control parameters that involved in industries to get a maximum trade-off between the maximum performance performance that can be achieved while minimizing all sorts of costs and possible expenditures. There are also specific variables by intake supply costs and labour costs that are more or less fixed and cannot be altered to much degrees. The use of Multi-optimization has been playing a crucial role in such industrial processes to reach the aforementioned targets of optimal design and performance constraints. For example the problems associated with labour and human elements have been largely mitigated in large corporations by substituting humans with robots wherever possible.

1.1.2 Multi-Modal Optimization

Optimization problems are often multi-modal that is having multiple set of valid solutions. Amongst this array of multiple valid solutions some could be locally good or globally good, and as such the goal of a multi-modal optimizer is to obtain the maximum possible sets of good solutions as opposed to a single set of good solutions.

Multi modal optimization [9] is very useful in fields like Engineering where due to physical design/cost constraints, the best solution to a problem is not always possible to implement. In such a case have access to array of valid optimal solutions other than the obvious best solutions can mean that the application can be quickly switched over to a different optimal solution. This preserves the best possible performance possible without burdening the designer to re-evaluate the optimization problem from scratch. Another advantage of using Multi-modal solutions is that by the evaluation of multiple solutions to the given problem, many hidden properties/relationships of the underlying optimization problem could be uncovered. This could be particularly helpful for various research aspirants interested in further exploring that particular domain. Furthermore multi-modal optimization not only locates multiple optima in a single run but also preserve their population diversity and hence multi-modal optimization algorithms can also be used for diversity maintenance techniques.

Use of classical techniques of optimization for evaluating Multi-modal optimization would need multiple restart points and with the desire of obtaining different solutions on each subsequent run. However there is no guarantee that different solutions will be obtained on each trail. Implementing optimization using Evolutionary Algorithms provides a natural advantage over classical techniques. The advantage of evolutionary algorithms are due to their population based approach, which are processed in every generational iteration, over the course of multiple iterations and if the multiple obtained alternatives are preserved at subsequent iteration, multiple good solutions can be obtained instead of a single best solution. This is totally in contradiction to classical techniques which always try to converge to a single best solution, or a sub optimal solution.

Evolutionary Algorithms is discussed in more detail in section 1.3.

1.2 DIFFERENT TYPES OF OPTIMIZATION TECHNIQUES

Optimization algorithms can be solved through

1.2.1 Calculus based methods.

Calculus is one of the fundamental and one of the most widely used components of modern mathematics today. The principles of modern calculus were independently developed by Sir Isaac Newton and Gottfried Wilhelm Leibnitz in the mid seventeenth century. The dependency on Calculus is so great in modern science that it is one of the first techniques that comes to mind while coming across evaluating any given problem.

1.2.1.1 The concept of derivative and gradient :-

Derivative of a function is defined as the rate of change of that function while in being compared to some standard function function. For example the rate of change of any given displacement with time is velocity, subsequently the rate of change of velocity with time is known as acceleration. As in using other examples, assuming perfect satisfactory conditions, within the linearity approximations of Ohms law for any given conductor, the rate of change of voltage with time is known as resistance, the vice versa being known as Conductivity. When applied to coordinate geometry the derivative of any function with respect to the other function is called as the gradient of the aforementioned function. For straight lines and planes the gradient remains the same, however for curves and complex 3D objects the direction of the curve doesn’t remain constant and therefore the gradient also continuously changes with the changes in the reference point.

1.2.1.2 Optimization using Calculas:-

Calculus can be used to evaluate simplistic optimization problems involving simplistic function that can be defined by simple polynomial expressions. The concept of maxima and minima is used here. The curve under examination is evaluated and it is observed that there are two specific points of the curve where the observed curve function experiences a minimum point and a point of maximum.

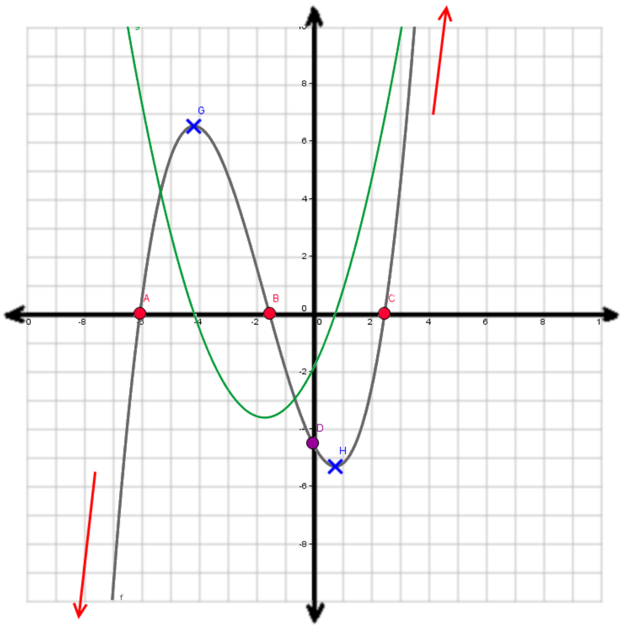


Figure 1.3:- Shows the points of maxima and minima in the two curves.

At the point of maxima and minima that at the point of peak and troughs, the gradient is zero. This concept is applied to find the optimal maximization or minimization of any given function. The function is first derived to obtain the points of either maxima or minima. Now if the point obtained is a maxima then from that point the curve may only descend that is if we take the derivative of the derivative (2nd order derivative) it will be a negative value, similarly for a minima the 2nd order derivative will always be a positive value since the curve can only rise after it's minimal value.

However it so happens that in most real life problems the problem function isn't just dependent on one particular variable. If there are two variable parameters involved, then partial derivatives are to be used.

It is to be understood that optimization by calculus is applicable only to the most basic problems and most real life problems cannot be effectively mathematically modelled appropriately to being optimized by using Calculus. Hence the next set of methods are more widely preferred.

1.2.2 Evolutionary algorithms

Discussed in detail in the next section.

1.2.3 Dynamic programming.

Dynamic programming[10] was developed in the 1950s by Richard Bellman and is a widely used optimization tool that has found wide applications in fields ranging from Economics, Control theory to aerospace designs. The working principle of Dynamic Programming is that it takes a complicated problem and breaks it down in simpler sub problems in a recursive manner. Finding out the optimal solutions to this set of this sub-problems recursively ultimately resolves the complexity of the parent problem solution. When some decision based problem is found unable to be split into smaller sub-components, the individual decision variables that occur over several different points in time are broken apart and evaluated recursively. It is to be explicitly stated that all the sub-problems that the main problems was classified into are all dependent and affected by each other. Dynamic Programming is also proficiently utilized in the field of computer science

Precisely stating, Dynamic programming is an optimization technique transforming a complex problem into a sequence of simpler problems, essentially . Dynamic programming provides a general framework for analyzing many problem types, where within this framework a variety of optimization techniques can be employed to solve particular aspects of a more general formulation. However certain amounts of creativity is desired so that a particular problem can be recognised and cast effectively as a dynamic program, and often little insights are necessary to restructure the formulation so as to resolve it effectively.

1.3 EVOLUTIONARY ALGORITHMS

1.3.1 The Theory of Natural Selection:-

The term Natural Selection [11] was popularized by Charles Darwin as an explanation for adaptation and speciation, as an explanation for his theory of evolution. Most of Darwin's ideas had been formulated based on the observations he made while aboard the HMS Beagle. In fact it was an en-route visit to the volcanic Galapagos islands where Darwin would make detailed observations that would form the crux of his theory of evolution later on. Darwin observed different species of Mockingbirds on each of the different islands he covered upon and he discovered that though each of the different observed species had many similarities, each of them had minute differentiations that would lead all of them to diversify and adapt to their respective home environments. This is where Darwin's idea of the theory of natural selection was born.

Charles Darwin defined natural selection as, *"principle by which each slight variation [of a trait], if useful, is preserved".* Natural selection of the differential survival of reproduction characteristics of different organisms due to the differences in their phenotypes. Natural variation occurs in every single organism on the planet. This occurs partially because random mutations can occur in arise in each individual species due to vast set of reasons ranging from changing climatic conditions to effect of UV rays due to decreasing ozone layers. This mutated genomes are hereditary and are often passed on the offspring populations which may experience further mutation themselves. What these mutated genomes do is change the trait of the living organisms living in the current environment. While this traits are usually minute there can be instances of drastic mutated offsprings that have wide physiological differences with parents. An example of that would be the rare Black Panther that occur due to a particular combination of recessive allele resulting in the secretions of enormous amounts of melanin coating on their furs giving to their rare black colour. Most of the times however this changes would be subtle and rarely noticeable. However this small variations add up with subsequent reproductions and it is observed that in a particular environment an individual organism with a particular trait continents but became islands due to the continental shifts. Both are full of individual species that usually have a counterpart in the mainland, and yet due to the passage of time have become completely different species. In Madagascar's case several of the older populace, that were separated in the split and ultimately did not fall victims to a newer set of environmental challenges that mainland Africa suffered, continue to thrive. gets a advantage over it's fellow organisms and it flourishes and thrives. Repeating this process over a period of millions of years and it is observed that individuals with particular genome variants survived and thrived while others with less variations perished. This is the crux of evolution and Natural Selection. Natural selection always dictates the survival of the fittest, the lesser perish. This is aptly described in the below attached figure where a sample population of bacteria were subjected to treatment of antibiotics. It is observed that Natural Selection only preferred those bacteria strains that were resistant to the antibiotic dose. Ultimately only those bacteria that were resistant to the antibiotic survived and thrived ultimately seeding the next population of bacteria all of which were resistant to that particular antibiotic.

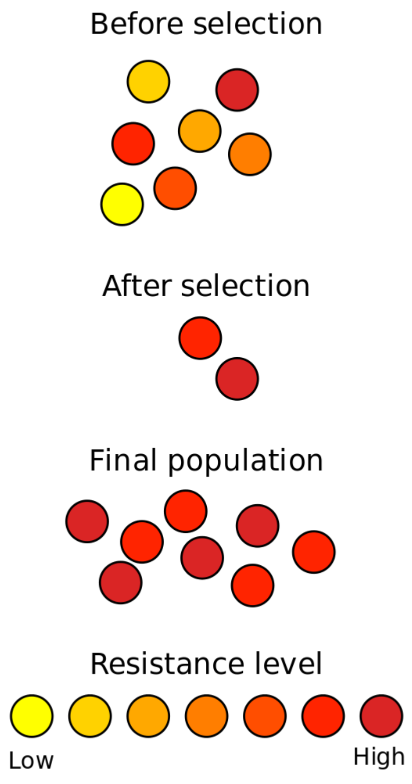


Figure 1.4:- Mixing of population of Bacteria with varying resistances to antibioics. Here shades of yellow and red are used to represent the varying degres of resistance to the antibiotic.

1.3.1.1 Types of Natural Selection.

- *Directional Selection* :- It is a type of Natural selection where a particular phenotype at the extreme end is preferred over other phenotypes. This results in the allele and traits to be dominated by the characterises of the selected phenotype over time.

- *Stabilizing Selection* :- This a particular type of Natural selection where the mean selected genome is selected from a non-extreme phenotype trait. This is the most popular type of Natural selection as effects of mutations are usually subtle and not drastic.

- *Disruptive Selection* :- The phenotype traits at the end are preferred over the ones at the centre.

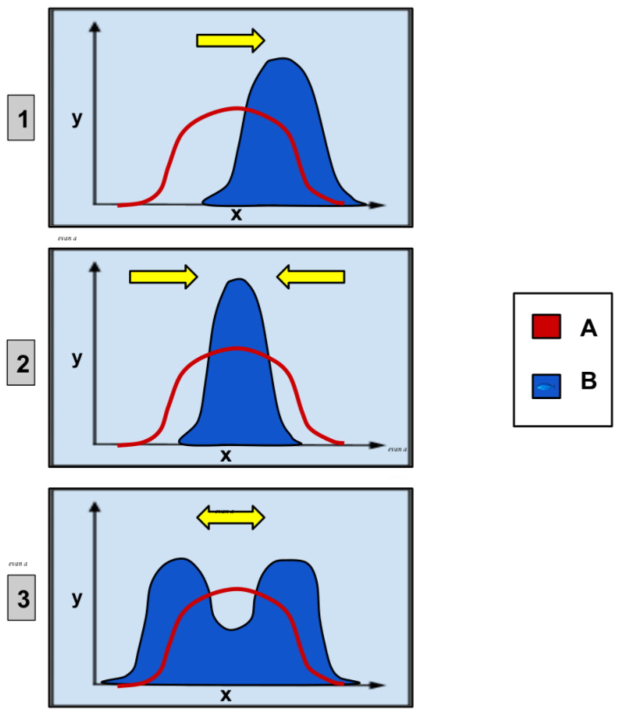


Figure 1.5:- Shows the different types of Natural selection in action. A is the parent population while B is the offspring. Case 1 is the result of directive selection, Case 2 the result of Stabilizing selection while Case 3 is formed due to disruptive selection.

Another example of the effect of Natural Selection, this however forced upon by human actions was the large scale release of soot as a by- product of the industrial revolution in eighteenth century England. Prior to the industrial revolution the normal light coloured winged moth was the dominant species of moth with the recessive traited black-winged moth forming only a minority of the total species. However due the enormous release of soot, all the trees were subsequently covered in large amounts of soot covering them all over in black. This gave a distinct camouflaging advantage to the then minority black-winged moths which suddenly started to multiply and flourish while the light coloured moth was now getting spotted easily and was hunted down by the predators in large amounts. It was only after the ‘Go-green’ initiatives of the 1950s that led to the reduction of the amount of industrial exhausts that light-winged moths started returning to their normal numbers.

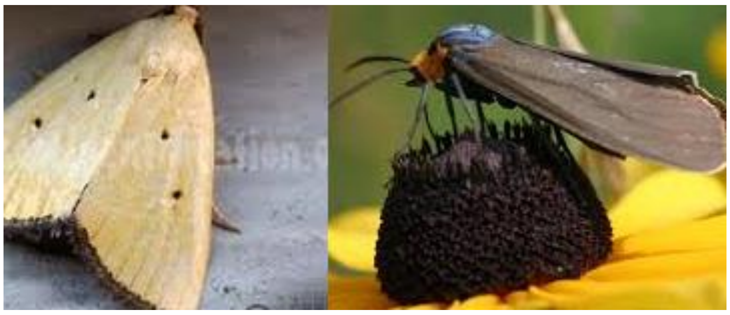


Figure 1.6:- Shows the light and black-winged moths. During the insutrial revolution the black moth gained a distinctive advantage and experienced a spurt in population numbers.

Natural selection is however different from ‘Artificial selection’ where the same process is however encouraged and a selective set of traits are breed over again and again for the purpose of human benefits. Darwin explained Artificial Selection as an example of selective breeding of crops that human beings do. A particularly vibrant example of that would be how bananas that used to be smaller and filled up to the brim with seeds, and how they were meticulously selectively cropped to get the modern banana, a much larger sterile fruit with large calorific value.

Another example would be the subtle modification of pigeon characteristics by Pigeon Fanciers to get different variations of pigeons.

The figure attached on the next page shows the subtle variations in pigeons observed.

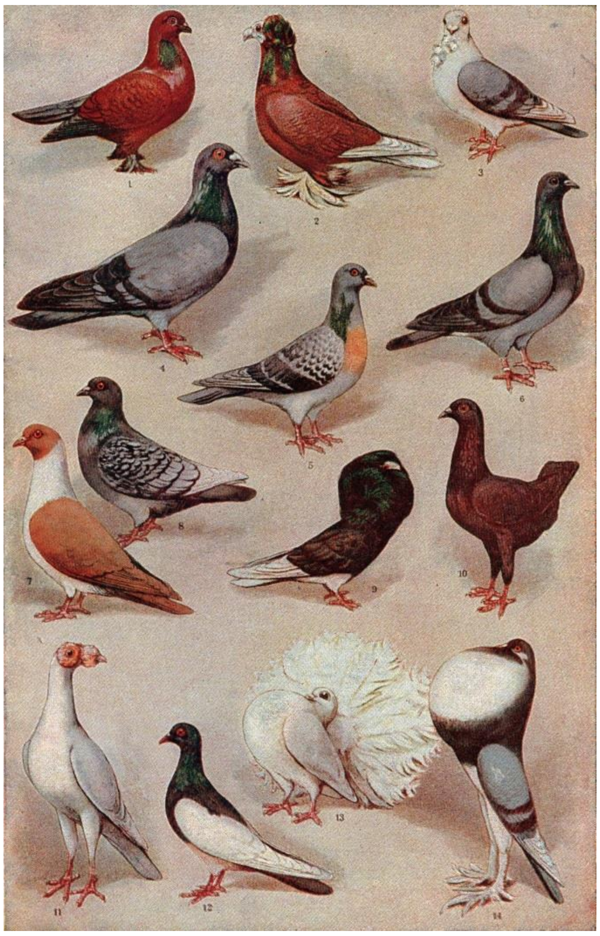


Figure 1.7:- An example of artificial selection is the creation of various pigeon like 1. Tumblers 2. Fantails etc by selective breeding.

In a nutshell, Natural Selection kills all those organisms that are deemed unfit for their environments by virtue of lacking those subtle phenotypic traits, while those more suited to that environment, that is the selected few who have the mutated specimen, survive and reproduce more prolifically. Thus, those individuals who survived long enough to breed successfully were selected by nature, which imposes a rough filter of predators and toxins on the organisms within it, and cuts many threads short. The value these individuals are maximizing is their number of descendents, or copies of their genes circulating in the gene pool.

1.3.2 Evolutionary Algorithms :-

As discussed in the previous sub-section, natural selection causes the survival and growth of only the fitness organisms. Evolutionary Algorithms [12] [13] (EAs) are direct modelling and application of the theory of natural selection and evolution into the real world world to optimize complex metahueristic optimization problems. Researchers are scientists alike were fascinated by the ingenuity and effectiveness of nature to solve several complex problems, it led to the formation of an entire different set of algorithms called Nature Inspired Algorithms (NIAs). Evolutionary Algorithms are a part of the NIA and they are widely used to evaluate complicated problems that would either be too cumbersome to solve by classical means or would be unsolvable entirely. The ingenuity of evolutionary algorithms is that, while classical methods try to converge on a particular optimal point by approximating their values closer and closer to that particular point resulting in a best possible solution, evolutionary algorithms try to make it's initialized population more and more compatible with the environment (specified conditions that require to be met). Here fitness isn't some objective function that requires to be optimized but more of a conformity to the environmental conditions. Evolutionary Algorithms tries to make it' s population vectors to conform those standards of environmental evaluations, thus instead of producing one best solution it produces a bunch of solutions all conforming to the required evaluating conditions. This is immense help in several practical solutions where if one optimal solution cannot be practically implemented, another solution can be immediately tried out without really compromising the performance parameters of the set-up.

In other words while different forms of EAs have their differences, the common underlying idea is that, given a population of individual solutions subjected to environmental compulsion, the principle of the 'survival of the fittest' takes over and the overall fitness of the solution improves with every passing generation. In general terms this is done, we are provided with a bunch of population solutions. Now the given quality function is used to measure the abstract fitness values, the more the better. Based on this fitness values the best of the original set/parent set are selected and seeded to form the next generation of solution sets also known as the offspring generation. This is done by the processes of Recombination and Mutation being applied to them. Finally the parent and offspring solutions are pitted against each other in a competition for survival and the one with the better fitness function is selected while the other is discarded. This is done over and over again iteratively until a desirable high quality of solutions with sufficiently high fitness values are obtained.

The basic attributes of EAs can be traced to two simultaneous processes that are employed in the working methodology of the algorithm. The first are the variations operations that lead to creation of novelty and uniqueness of the generated solutions. They are mainly the operations of Recombination and Mutation. They are responsible for diversifying the solution by generating unique and novel solutions. The second process would be the art of selection. By putting the parent and offspring in a competitive battle and selecting the fittest solution of the lot improves the overall quality of generated solutions tremendously. It is to be noted that during selection the fittest of the solutions have a higher probability of getting selected over a solution with lower fitness values. Nevertheless even the weaker solutions also have a chance to survive or even be a parent. Since for recombination part the choices of the selected parent solutions is purely at random. Similarly for the mutation the individual element chosen to be mutated or portion of the piece to be replaced are all chosen in normal. This ultimately improves the diversity and novelty of the solutions generated at every generation.

In general the steps used in a EA can be represented by the pseudo-code shown in the below figures :-

```
BEGIN
        INTIALISE candidate population solutions;
        EVALUATE fitness;
        REPEAT UNTIL (Terminating condition acheived)
           DO
           SELECT parent solution;
           RECOMBINE pair of parent solutions;
           MUTATE resulting offspring;
           EVALUATE fitness;
           SELECT next generational solutions;
END
```

Figure 1.8:- The general scheme of an evolutionary algorithm represented in a pseudo-code.

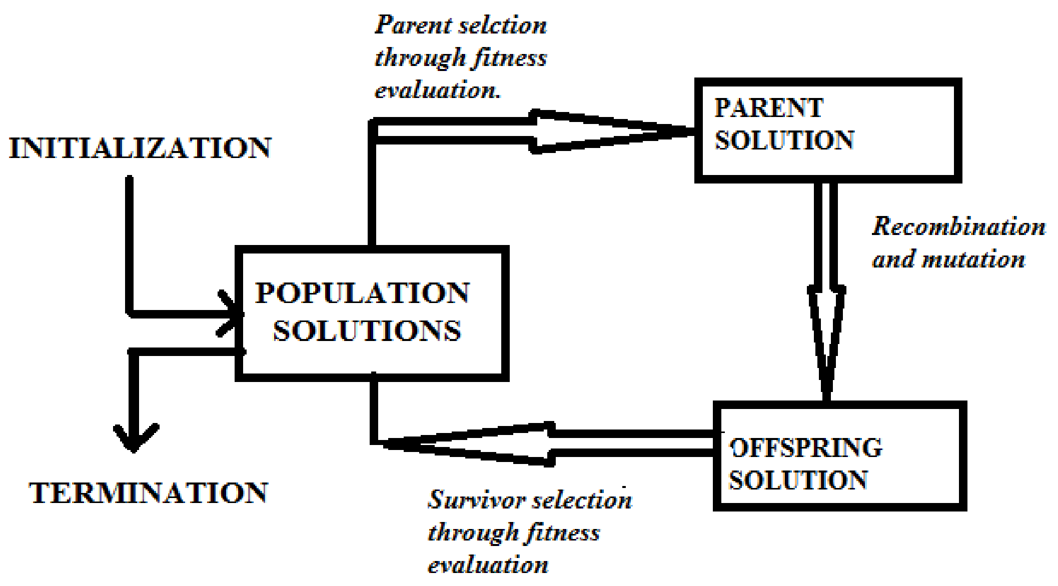

Parent selction through fitness evaluation.
PARENT SOLUTION
INITIALIZATION
Recombination and mutation
POPULATION SOLUTIONS
OFFSPRING SOLUTION
TERMINATION
Survivor selection through fitness evaluation

Figure 1.9:- Figurative representation of EA.

1.3.2.1 Steps in Evolutionary Algorithms

EAs follow a sequence of steps which have elaborated individually as follows :-

- Representation
- Fitness function or Evaluation function
- Population representation
- Parent selection mechanism
- Variation operators (Recombination and Mutation)
- Survivor selection mechanism

1.3.2.2 Representaion

The first step in Evolutionary Algorithm is link up the given real world problem to the parametric specifications of the EA. That is to bridge the real world problem to the context of the problem solving space where the evolution is set to take place. The possible problematic solutions in the real world problem are referred to as phenotypes whereas upon encoding the equivalent output in EA is known as the genotype. Representation is basically mapping of the phenotypes onto the genotypes that are set to represent the phenotypes in the EA workspace. It is to be understood that that the phenotype space is likely to be vastly different from that of the genotype. A good solution that is a good phenotype is obtained after decoding the best obtained genotype after the termination of the algorithm. In the context of EA terminology the word candidate solution can be used for representing the phenotype space whereas genotypes are typically referred as chromosomes.

1.3.2.3 The Fitness Function

The task of the evaluation function is specify the conditions that the solution must conform to and as such it represents the requirements that must be met with. It is the basis of further selection and facilities improvement, rather to put it more aptly it represents what improvement actually means. From the perspective of the real world problem, it actually directs what improvements must be made into the genotypes.

In order words it is basically the standard that assigns a quality measurement index to the various genotypes.

For maximization problems the evaluating function is referred to as the fitness function where the concept of ‘the more the better’ is appilicable.

For minimization problems this is known as the objective function.

The various fitness functions used in context of this thesis will be elaborated further in Chapter 3.

1.3.2.4 Population representation

The population is basically the representation of possible solutions, that is it is a multi-set of various genotypes. In most types of EA applications, the population size remains a constant. The population is basically the component that evolves and not the individual chromosomes. The representation of population depends on the type of EA used. In simplistic EAs it can be specified by defining the number of individuals populating it. In sophisticated EAs, the population may have an additional spatial structure representation describing either a distance measurement or the existence of neighbourhood. The concept of neighbourhood will be further touched upon in chapter 3. In such cases additional specifications have to be given to fully described the aforementioned population. It is to be noted that unlike the variation operators (recombination and mutation), the selection operators act on the population as a whole. Variation operators act on either a single individual(mutation) or on a pair(recombination). Selection operators however work at the entire population level and give an output relative to the entire population.

The diversity of a given population is said to be the measuring yardstick of the type of solutions present in the population. The diversity of any given population may be specified by the parameters such as the numbers of fitness functions used, or specifying the numbers of phenotypes and genotypes used.

1.3.2.5 Parent Selector mechanism

This mechanism is tasked with sorting out the individual elements based on the evaluation with the fitness function. This is done so the most fit individuals are given a preferential priority in becoming parent solution.

A parent solution is one that is allowed to seed offspring solutions through being operated by the variation operators. Together with the survivor selection operator mechanism the parent selector mechanism is responsible for improving the net quality of the subsequent solutions with every passing solutions. The selector mechanism is totally probabilistic with individuals with highest fitness values getting the most chances to seed an offspring. Nevertheless some chances are also provided to weaker individuals to prevent the overall search operation from becoming too greedy and getting stuck into a local optimum.

1.3.2.6 Variation operators

Variations operators are tasked with the creation of newer solutions and thus improve the overall diversity of the population. There are two variation operators namely :-

- Mutation :- Mutation is problem specific unary hueristic variation operator when applied to a parent genotype delivers a slightly modified/ mutated output that is referred to as the child or offspring genotype. A mutation operator is always stochastic in nature, that is the offspring formed due to it is dependent on a series of random choices. It is to be noted that implementation of any variation operator is akin to stepping in on a new point in the search space. By virtue of modus operandi mutation can virtually guarantee that every point on the search space is connected. This is because mutation and recombination working in tandem act as a linking pathways between all generated parent and offspring solutions. This is primarily the reason why it is believed that given enough time a EA will always discover the global optimum of any given search space.

- Recombination :- Recombination or crossover is binary variation operator that merges the genetic information between any given two parent solutions to form one or two new offspring solutions. Like mutation crossover is also a stochastic operator that is the choice of parents that recombine and what portion of their information recombine is totally dependent on random drawings. The working principle of Crossover is that from a set of two parents with different but desirable characteristics we can easily obtain an offspring solution having both the desirable quantities.

1.3.2.7 Survival selection mechanism

After the variation operators have been used to create new solutions the survival selection mechanism is used to determine the genotype that would be allowed to pass on into the next generation. This selection is again based on the fitness function evaluation and the comparision is made by the pitting the newly generated offspring against it’s parent genotype. Based on fitness comparisions the better genotype is allowed to pass into the next iteration. It is however to be noted that unlike the parent selection mechanism the survival selection mechanism is not stochastic but highly deterministic. The selected solution that is allowed to pass is always strictly dependent on it’s fitness values.

Beside the aforementioned steps in Evolutionary algorithms there are two surplus steps namely the initialisation and the termination criterion. The initialization criterion is mostly simple and the first random population is formed by seeding random initial individuals, though there are provisions to generate an initial population with high fitness values if specific heuristic conditions are specified. The terminating condition is however more problem specific in nature and requires closer regulation.

Mostly the algorithm is terminated when either the maximally allowed CPU time elapses or for a given amount of time the fitness improvement stabilises are no major improvement is seen

1.4 DIFFERENT TYPES OF EVOLUTION ALGORITHMS

The most common types of Evolutionary Algorithms are as follows:-

- Genetic Algorithms (GAs) [14] [15] [16] :- The most popular type of Evolutionary Algorithms. The general principle remains the same.
  From a population of randomly initialized population at every passing generation the fitness of the generated solution is first evaluated, the fitness being measured is by an objective function and then the best valued solutions are stochastically then selected for being subjected to variation operations. Crossover is preferred first here though and the resultant solutions are then subsequently used over in the next generation until the algorithm converges to an optimal solution.
- Differential Evolution (DEs) [17] [18] [19] :- Differential Evolution is a newer form of EA that is gaining huge popularity amongst various researchers due to it's numerous benefits and numerous derivatives are being released daily. DE makes no assumptions regarding the data set fed to it to be optimized, it can handle enormous amounts of data as well being able to optimize noisy discontinuous real world problems. It is a fast converging algorithm that optimizes a given problem by generating newer solutions sets from the existing solutions using it's simple formulae, and then preserving the solution sets which have the highest fitness values. In DE mutation is given priority over crossover.
- Genetic Programming [20] [21] :- It is a technique of generating programs that behave analogous to the process of the evolution. A random unfit solution set is first initialized and then acted upon by a set of steps that closely model the process of natural selection. After passing through the variant operators the resulting population is recursively applied with other solutions to obtain a newer set of populations that are usually more fit than the preceding populations.
- Evolutionary Programming [22] :- It is similar to genetic programming, the difference being that the main program structure to be optimized is kept fixed only changing it's numerical parameters. Uses mutation as it's main variation parameter.
- Gene Expression Programming :- A variation of the EA it is used to design complex programs that form complex tree like structure that

change their shapes, sizes and compositions based on their abilities to learn and adapt.

- Evolution strategy :- A natural problem dependent representation is used in combination of evolutionary algorithm applied in a loop. One execution of the loop is termed as a generation. Uses Muation and Selection as major search operators.
- Neuroevolution :- It applies Artificial Intelligence in combination with evolutionary algorithms to create Artificial Neural Network (ANNs) . Has a wide area of applicability.
- Learning Classifier Systems (LCS) :- LCS is a rule based machine learning methods that combines evolutionary algorithm with a learning component performing either supervised learning, reinforcement learning or unsupervised learning

In this thesis our focus would be on Swarm Intelligence a similar and alternate technique to EAs. Swarm Intelligence is discussed in detail in chapter 2.

1.5 THESIS OVERVIEW

Optimization of multi-dimensional and multi-modal problems can be realistically and practically evaluated using Nature Inspired Algorithms(NIA) such as Evolutionary Algorithms and Swarm Intelligence. The advantages of NIAs is that rather than converge on a specific solution for the given problem, NIAs give multiple sets of solutions that maybe individually different but all are compatible with the prerequisite conditions specified by the user.

Swarm Intelligence is a modelling of collective intelligent behaviour observed amongst social species like birds , fishes and insects, into optimization algorithms to effectively deal with complex and noisy multivariable and multimodal problems. One of the most popular Swarm Intelligence based algorithms is the Artificial Bee Colony(ABC algorithm). The objective of these thesis is to show the improvements in the explorative search aspect in the ABC Algorithm by using a neighbourhood based mutation variable. Through this study we try to show the benefits of adding separate local and global fitness evaluation parameters to the ABC algorithm. This modifications suggested in these thesis helps in making the ABC more versatile due to the offered flexibility in tuning the algorithm for better exploitative or explorative searches through the variation of the new parameter, the weight function, being introduced here.

2 AN OVERVIEW OF THE ARTIFICIAL BEE COLONY ALGORITHM

2.1 ABOUT SWARM INTELLIGENCE

2.1.1Swarms

Swarms or rather the act of swarming is a phenomenon displayed by organisms of similar size by aggregating together *en masse* around the same spot or perhaps moving or migrating in some particular directions. The motivation for such actions may range from security in large numbers to large scale long distance migrations in herds of thousands over intercontinental distances. Swarm behaviour across multiple species given a specific set of names. For example flocking refers to swarm behaviour of birds, herding is referred to as herds of tetrapods (reptiles, amphibians, mammals), schooling is the name given of swarm of fishes. Even phytoplanktons are found in enormous swarms known as blooms.

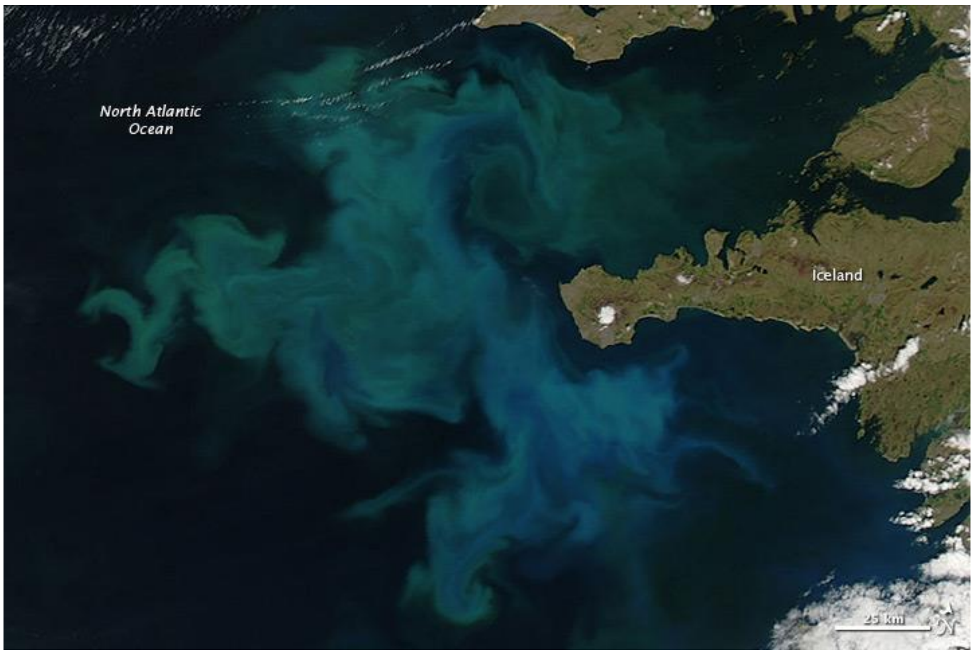


Figure 2.1:- Figure represents a huge bloom of phytoplanktons of the coast of Iceland.

However swarm behaviour isn’t explicitly limited to natural objects as various artificial and inanimate entities also experience swarm behaviour namely swarms of robots, or a swarm of stars. In a broader sense a swarm behaviour may be expressed as the collective motion of a large number of self propelled elements. Swarm behaviour is elegant to observe as there is no central coordination between the participating entities and yet subtle alternations in individual participants when added up lead to massive alterations in the collective products. It is almost like the entire herd though individually capable of independently acting on it's own, each contributes to a much larger cause, the result of which is a form of collective intelligence . Researchers across various fields were highly fascinated by this and a new field of study came into being known as swarm intelligence.

2.1.1.1 Swarm Behaviour observed in nature.

- Insects :- Researchers have been long fascinated by the behaviour of social insects like ants, bees and termites that often live together in large organized colonies. It has been observed that most insect colonies are self regulated and individual insects do not act anything out of the ordinary and yet the colonies are surprisingly well organized. It so happens that the entire coordination and cooperation across the group happens just as a consequence of the way the individuals act.
  The interactions in between the individuals dwelling in the colony can be absolutely trivial such as following the trial of pheromones left by another ant, and yet when summed together and interactions observed as a whole, the cumulative consequences of such actions can be used to solve problems of enormous complexity.

Figure 2.2:- Shows a fire ant colony.

Figure 2.3:- A functional Killer bee colony. Killer bees are regarded amongst the most dangerous animals in the world.

Observation and modelling of swarm behaviours in social insects like bees and ants, to real world problems have given rise to highly popular algorithms like the Ant Colony Optimization and the Artificial Bee Colony algorithm.

- Birds and fishes :- The primary reason for birds engaging in a flock behaviour is large distance migrations. Several species of Geese travel enormous distances often across intercontinental distances. The primary motivation for migration is food and extended breeding seasons. Long winters in the poles often lead to scarcity in food supplies and thus several birds migrate to the opposite poles where it is summer. The birds take advantage of the aerodynamic benefits by flying in the V-shape. This conserves energy while also helping in watching out for predators.

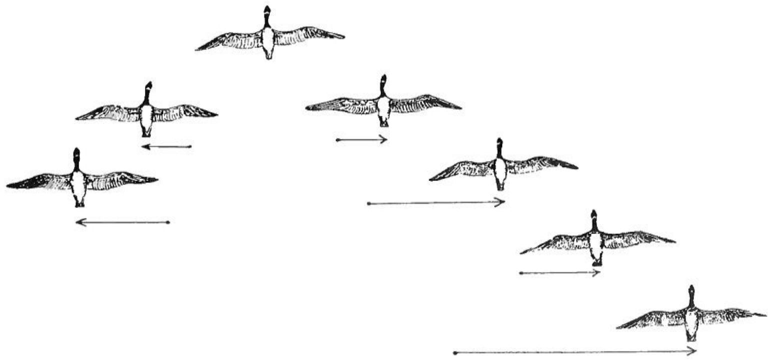


Figure 2.4:- Shows a flock of migratory geese flying in the V shaped formation.

Another advantage of sticking together in large numbers is the obvious protection in large numbers against predators. Sticking together in large numbers is beneficial for survivability as the probability of survival increases. Examples of such behaviour would be in auklets and starlings.

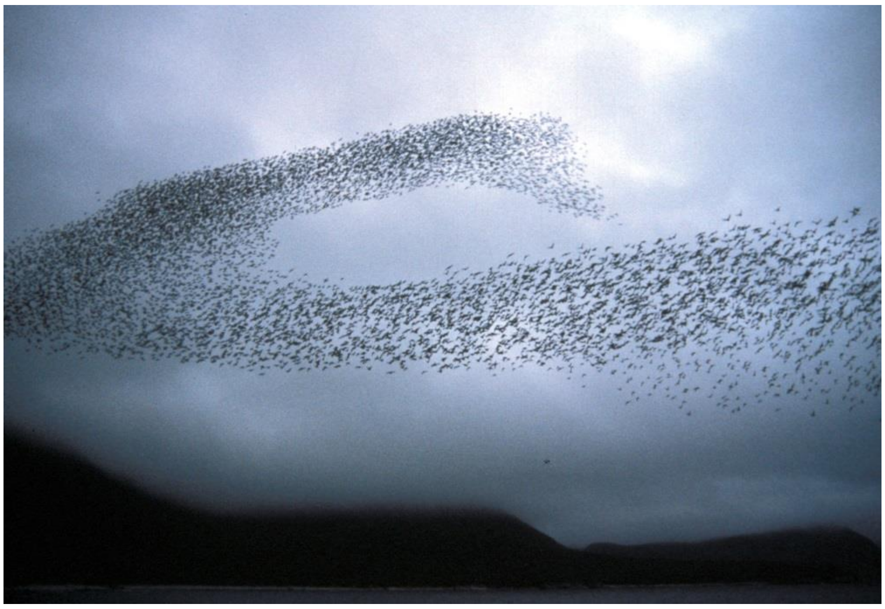


Figure 2.5:- Flock of auklets in defensive stride and on predator watch.

Figure 2.6:- Flock of starlings in aggressive posture while migrating.

Similar to birds, shoals of fishes often stick together for the reasons odds of better survival in larger numbers, better foraging for food and better chances of breeding.
The study of breeding of birds and fishes were the main motivations for the creation of the highly popular Particle Swarm Optimization.

- Human swarms :- Even human beings can exhibit swarm behaviour in the form of crowd behaviours

2.1.2 Swarm Intelligence :-

When a swarm of body particles interacting freely with each other collectively form a connected network that can be used for evaluation of complex problems, it is known as Swarm Intelligence[23]. As per Bonabeau [24][25]Swarm intelligence is "any attempt to design algorithms or distributed problem-solving devices inspired by the collective behaviour of social insect colonies and other animal societies". Swarm intelligence is a NIA mimicking the intelligent behavioural characteristics of social colonies of insects, birds fishes etc. These colonies are built around huge number of individual animals, all contributing and cooperating for the preservation and growth of the colony. The advantages of engaging in societal behaviours are multi-fold. Indeed these are chief reasons for the rising popularity of swarm intelligence amongst researchers. Highly complicated and multifaceted problems can eliminated through use of of skilled swarm intelligence.

In a swarm exhibiting intelligent behaviour two concepts are absolutely fundamental so as to maintain the structural and functional identity of the swarm. They are self-organization of the swarm, and the division of labour amongst the various swarm components. [26]

- Self-organization can be defined as " a set of dynamical mechanisms, which result in structures at the global level of a system by means of interactions amongst it's low level components". The basic laws of interactions between the individual elements are established by this mechanism of Self-organization. This set of well defined rules ensures that the local level interactions between the individual components are totally independent and unaffected by the happenings in the global pattern. There are four basic characteristics and they are classified below :-

1. Positive feedback :- The simple thumb rule that promotes the creation of convenient chain of structures in the establishment of the network. For example the practical examples of laying of food trail by an individual ant or the dance of a forager bees while on there own may seem trivial but these small actions are the starting points of the formation of a swarm of ants/bees on the lookout for food. They are examples of positive feedback.
2. Negative Feedback :- The negative feedback mechanism is employed to ensure swarm stability by countering the positive feedbacks to some extent. This is extremely important to avoid conditions such as over-population of foragers being tasked onto a food source leading to consequences such as food exhaustion, crowding and competition around the food source.
3. Fluctuations and random switches :- Simple and often seemingly trivial actions like random switching over the tasks, random lookout for newer food sources are necessary for maintaining the novelty and creativity of the swarm based intelligence. This random behaviour is often crucial as it enables in the discovery of newer solutions.
4. Mutually tolerance :- In general self-organization demands the there should be an element of mutual tolerance between the individual components in the swarm. This is to facilitate cooperation and coordination throughout the swarm, this enables the individual results and interactions and results being shared amongst the various swarm members. Ultimately this improves the swarm efficiency.

- Division of Labour:- The specialization of different tasks which are to be performed simultaneously by specialized individuals throughout the swarm is known as the division of labour. This methodology of simultaneous task completion by the coordinated efforts of specialized individuals has been found to be much more efficient than the completion of the task in a chain of sequence by unspecialized individuals. Division of labour also enables the swarm to adapt to sudden alternating conditions in the search space.

2.1.3 Some popular applications of Swarm Intelligence

1. Particle Swarm Optimization [27] :- Introduced by Eberheet and Kennedy this is a highly popular stochastic search algorithm modelled on the behaviour of a flock of birds or shoal of fishes. PSO computes an optimization problem by iteratively trying to improve a candidate solution based on the evaluation of quality. It solves a problem by initializing candidate solutions and then moving these candidate solutions over the search space based on simple mathematical evaluation of the position and velocity of the particle solutions. Each particle's movement is influenced by it's local best solution, but they are also influenced by the global best solution, which is basically updated by better positions by other particles. This basically moves the swarm towards the best possible solution. PSO is basically a metaheuristic search based algorithm that makes few or no assumptions regarding the problem at hand and act on a huge number of candidate solutions.
   However PSO does not always guarantee optimal solutions to a given problem.

2. Ant Colony Optimization [28] :- Ant Colony Optimization was originally proposed by Marco Dorigo, is another popular application of swarm intelligence that is modelled based on the behaviour of ant colonies. The ants (simulation agents) basically locates any optimal solutions by moving through a parametric search space based on the real movement of ants. Real ants utilize pheromones to convey information to each other, when a random ant detects a newer path of food sources and lays a trial of pheromones for other ants to explore and exploit. As more and more ants explore these newer pathways, the pheromone content is further enhanced and more and more ants converge on this pathways. In ACOs the simulated ants record their respective positions and the quality of their discovered solutions, as more and more iterations pass by, the higher qualitative solutions get more and more traffic and the solution converge to optimal values. Like PSO, ACO is also a metaheuristic algorithm and is capable of optimizing large areas of search space.

2.2 INTRODUCTION TO ARTIFICIAL BEE COLONY ALGORITHM

Artificial Bee Colony optimization is one of the most widely used and popular swarm intelligence techniques proposed in October 2005 by Dr. Dervis Karaboga[26][29].Since it's inception this algorithm has been in wide use over the scientific community. Furthermore, a number of reserachers have subsequently modified this algorithm and applied it to different areas of engineering and science to solve various complex problems. The simplicity and effectiveness of this algorithm lies in it's use of minimum number of control parameters and quick resolution of it’s problem variables

The strategy of ABC is clear-cut and it’s swift use of population based stochastic search strategy makes it a highly preferred choice amongst researchers delving in the area of Nature Inspired Algorithms (NIAs) for optimization. Karaboga’s proposal was to the model the food foraging behaviour of the honey bees into a simple and effective optimization strategy. Similar to other population based algorithms, this algorithm also starts with it’s population of potential solutions. The individual food sources for the honey bees are the representation of a possible candidate solutions. The quality of the food source, that the nectar content of the food source is given by the fitness function. The number of food sources is analogous to the given population size. There is also a parameter limit introduced that serves a limiting condition for potential food exhaustion and the signal for further exploration of food choices. The performance of the ABC algorithm is highly dependent on the size of it’s chosen initial population. If the population is kept on increasing then, after reaching a certain limit the performance of the ABC deteriorates considerably. The limit parameter which is function of the population size and problem dimension can also be considered to an indirect contributor to the performance parametric of the ABC algorithm. Similarly problems of extremely high complexity and dimensionality can also lead to the deterioration in the performance of the ABC algorithm. Regardless of it’s limitation, the ABC algorithm since it’s inception has been highly popular because of it’s robust nature, it’s ease of application and due to it’s minimal control parameters.

2.3 LITERATURE REVIEW

2.3.1 Behaviour of Honey Bee Swarms :-

The minimalist model of a swarm of honey bees foraging for food sources consists of two essential components, the food sources themselves , the forager bees who are further classified into employed and unemployed foragers[26][29]. All of which are elaborated below:-

- The Food Sources:- The food sources are necessary for providing nutrition for the bee hive. The feasibility of accessing a particular food source plays an an important role in determining the quality of the food source. Generally speaking the value of a food source for a honey bee is dependent on many factors such as proximity to the hive, it’s nectar content, and the ease of accessing it’s nectar. However in ABC, for the sake of simplicity, the food quality is represented by a single quantity.

Figure 2.7:- A bee scouting for food on a flower.

- The Forager bees :- The Forager bees are of two types the employed and the unemployed foragers.

1. Employed forager bees :- These are the bees that are associated with a particular food source. They are currently at the food site exploiting the food reserves. They carry over the all the required information about a particular food source, it's distance and direction from the bee hive, and the feasibility of it's extraction. Figure 2.7 is the example for a scout bee that becomes an employed forager on finding nectar.
2. Unemployed forager bees :- These are continuously on the look out for food sources to exploit. They are further classified into two types ***scouts*** and ***onlookers.*** The scout bees are on the prowl always looking out for newer food sources to exploit. The onlooker mostly stay at the hive waiting the employed foragers to transfer the food information to them. These bees then become follower bees of the employed forager. A scout can become an employed forager too if it is able to find a suitable food source worth exploiting. Scout bees are usually rare averaging 5%-10% of the total beehive strength.

The exchange of food information amongst the various members of the bee hive is the single most important occurrence in the formation of the collective knowledge. All communication between the bees in the bee hive takes place in a particular sequence of dance, called the ***waggle dance***. [26][29] The waggle dance is a highly specific manoeuvre, consisting a series of waggles and curves pertaining the information regarding the food sources. The more the number of waggles the greater distance of the food source. The direction of the waggles is always conveyed with reference to the sun outside the hive, and with respect to the top direction when inside the bee hive. Higher the food quality, more prolonged is the waggle dance.

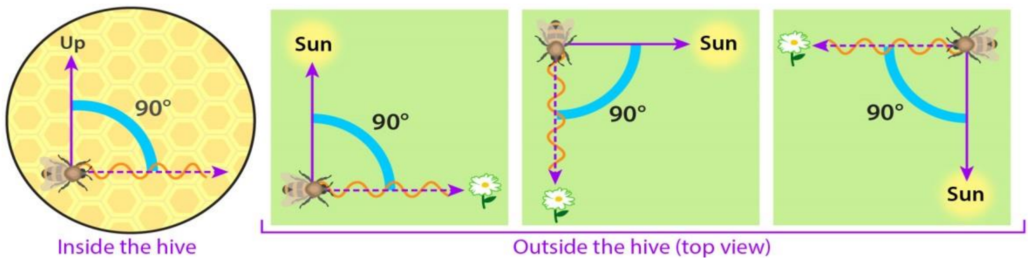


Figure 2.8:- The directionalities of waggle dance shown in figure

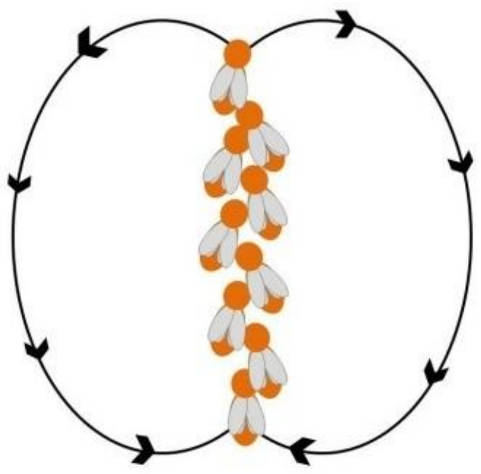


Figure 2.9:- shows the complexities of the waggle dance. More the wiggles further is the food source.

The information conveyed in the waggle dance is available to every onlooker bees in the hive. Usually there are multiple waggle dances occurring simultaneously inside the hive, an onlooker can observe multiple waggle dances and then decide to choose the food source that appears the most profitable to her. There is an obvious greater chance that the food sources with greater profitability will be taken up by more number of onlooker bees.

The information regarding the food sources is passed on by the employed foragers bees through waggle dance. To understand this in detail, let explore this process in a bit more detail. Let us assume that there are three potential food sources A, B & C. At the very beginning, all bees start off as unemployed foragers. The bee under observation has no knowledge about the whereabouts of the food source. It faces a choice :-

1. Either it starts of as a scout, scouting for potential food solutions due to some internal motivation, or due to some external clue which maybe the exhaustion of current food supplies or overcrowding or some other reasons.
2. Otherwise it observes the waggle dance of an employed forager and becomes a recruit and then converges on the food source.

After reaching the food source the bee will absorb some nectar that begin exploiting the food source, and then will memorize the location and direction of the location from it’s hive. The bee now is an employed forager when it returns to the hive, it drops off the nectar and is now faces with three choices.

1. It abandons the food source and returns to the dance floor.
2. It goes the floor and dances, and in turn recruits other potential bees and leads them to the food source.
3. It continues exploiting the above found food source without recruiting any new bees.

It is to be noted that not all of the available worker bees starts foraging a discovered food simultaneously. Experiments have shown that if only a small fraction of the available workforce is foraging, the rate at which the present bees starts forages increases goes up and vice versa. Furthermore when the food source is exploited to a particular extent , scout bees fly off in search in of newer food sources. This is the beauty of the collective intelligence exhibited by the swarm that is employed in the ABC algorithm.

The self organization in the swarm of honey bees is deciphered as :-

1. Positive feedback :- Greater the number of food sources, greater is the number of onlooker bees visiting them.
2. Negative feedback :- The exploration of the food sources is stopped by the bees.
3. Fluctuations :- Random searches are carried out by scout bees and often discover newer food sources.
4. Mutual interactions :- Through waggle dances bees are able to communicate with other bees regarding the food sources.

The division of labour is as follows :-

1. The queen bee and the drones are only concerned with breeding and take no part in gathering food.
2. The Defender bees :- They are tasked with the protection of the bee hive
3. Th worker bees :- These are the bees that take part in nectar gathering. These bees act as employed, onlookers and scouts. Our interest lies in these bees only.

2.3.2 Artificial Bee Colony Algorithm (ABC)

The ABC algorithm is modelled on a colony of worker bees classified into three namely, the employed bees, the onlooker bees and the scouts.

Employed bees are those bees which visit a particular food source that it has previously been to, or goes off to as a recruit after following the waggle dance of some other bee. The onlooker bees can be found on the dance area observing the dance of other employed bees, ready to make a decision of choosing to visit a particular food source. Scout bees randomly scout out for newer food sources. In the context of the ABC algorithm the entire population is split into half, one half consisting of employed bees and the other half consists of the onlooker bees. The number of potential solutions locations/ food sources is equal to number of employed bees. In other words, every potential solution has an employed bee attached to it. If by any chance the food source attached with the particular employed bee gets exhausted, the bee becomes a scout and randomly scoops out newer possible location of the solution/food sources.

The modus operandi of the ABC algorithm is as follows, every cycle of the iterative search is comprised of three steps:-

1. Sending the employed foragers onto the food sources/potential location of solutions and measuring the quality of the nectar (fitness of the solutions) .
2. Selection of a particular solution/food source by an onlooker bees after observing the quality of all available food sources as conveyed by the employed bees. The quality of the food sources is calculated again by the onlooker bees.
3. Finding the scout bees and taking advantage of the possibility of newer food sources being found by them.

2.3.2.1 The initialization phase

There are chiefly three constraints in a ABC algorithm namely, the number of food sources, the limit that is set so to define the attempts of retrial to exploit a food, following the expiry of which the food source is rejected. The third constraint is the number of iterative cycles that the algorithm is allowed to execute before being terminated. It has already been mentioned previously that the number of employed and onlooker bees are kept equal to get best results.

The initial population consists of food sources where each food sources is symbolized by a D Dimensional vector x_i (i = 1,2….NP) where NP is the number of food sources. The food sources are initialized by the following equation

$$x_{ij} = x_{\min j} + rand(0,1) * \left(x_{\max j} + x_{\min j}\right) \ldots (2.1)$$

2.3.2.2 Employed bee phase

This is the second phase of ABC algorithm. In this step the existing food sources are replaced with innovative newer food sources having better value of fitness calculated with the help of a fitness function. This newer solutions are represented as

$$v_{ij} = x_{ij} + rand(0,1) * (x_{ij} - x_{kj}) \ldots (2.2)$$

Where j ∈ {1,2….D} and where k ∈ {1,2….NP} and k should be chosen that k≠i . This is done so as to ensure definitive improvements in the newer candidate solutions.

2.3.2.3 Onlooker Bees phase

The third phase of ABC algorithm is the onlooker bee phase. In this phase the information is transferred from the employer bees to the onlooker bees in form of fitness. Every food source is judged based on it's evaluation based on fitness function. The fitness function is calculated on the basis of an objective function. Ultimately the highest quality solutions are selected based on the fitness function and selected based on probability which is calculated based on the fittest set of solutions. Thus the onlooker bees is flooded with the best solutions based on fitness evaluation. The received solution is again modified by the onlooker bees and further checked for the fitness value.

Calculation of fitness function :-

Assumption :- Used objective function is greater than zero

$$Fitness\ function = \frac{1}{2*(objective\ function)+1} \ldots(2.3)$$

where objective function is the performance evaluation function used

The probability selection is determined by selecting out those solutions which have the maximum fitness function amongst the total number of solutions available

$$P_{ij} = \frac{fit_i}{\sum_{i=1}^{NP} fit_i} \ldots(2.4)$$

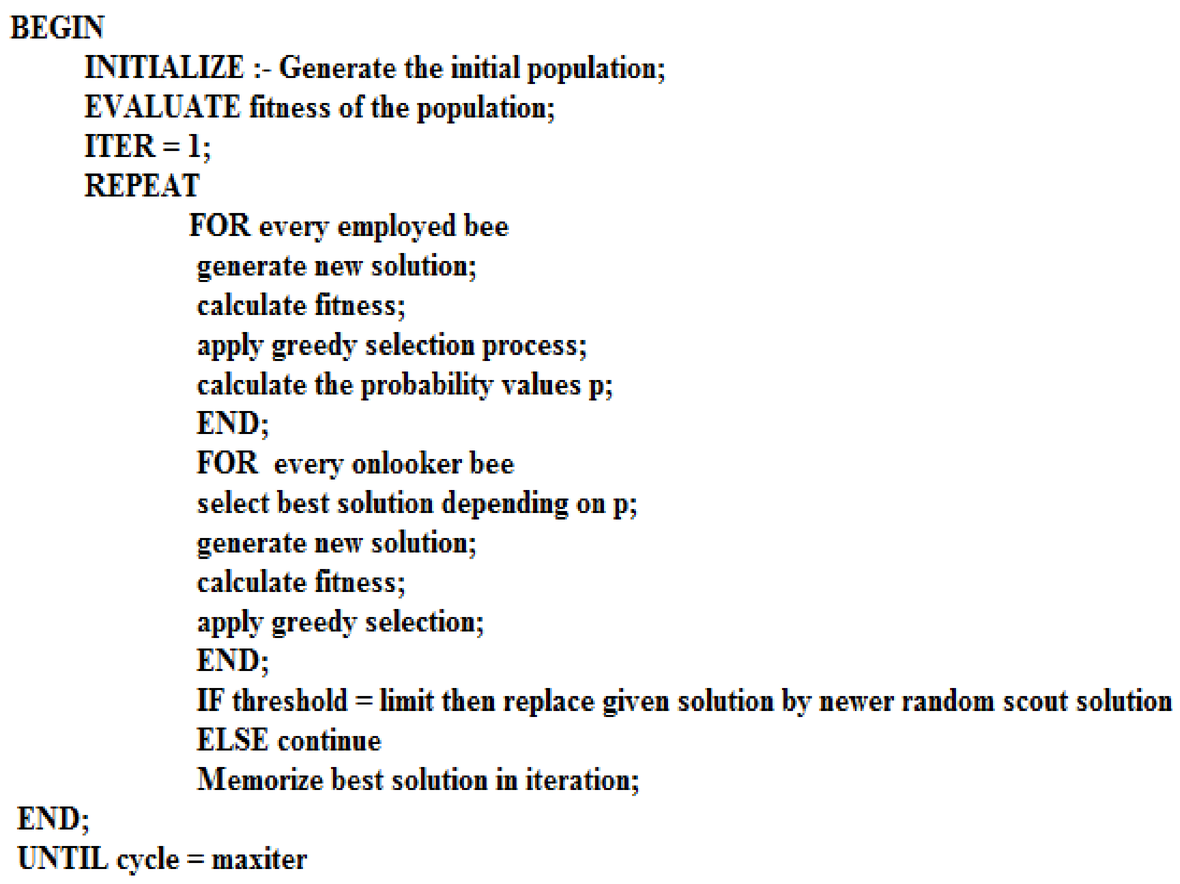
2.3.2.4 Scout Bee Phase

The final phase is the scout phase where there is an attempt to foster newer random solutions. There is a threshold parameter that is defined at the start of the algorithm. The threshold parameter counts the number of trials that has taken place, and whenever and wherever the value of the threshold exceeds or equals the value of the limit, the scout phase is started and the threshold value is recalibrated to zero. The Newer food sources are generated as:-

$$x_{ij} = x_{\min j} + rand(0,1) * (x_{\max j} - x_{\min j}) \ldots(2.5)$$

After the scout bee phase is completed, the algorithm conditions are evaluated and the optimized solutions are displayed after the algorithm terminates.

2.3.2.5 ABC Pseudo Code



```
BEGIN
          INITIALIZE :- Generate the initial population;
          EVALUATE fitness of the population;
          ITER = 1;
          REPEAT
                    FOR every employed bee
                     generate new solution;
                     calculate fitness;
                     apply greedy selection process;
                     calculate the probability values p;
                     END;
                     FOR  every onlooker bee
                     select best solution depending on p;
                     generate new solution;
                     calculate fitness;
                     apply greedy selection;
                     END;
                     IF threshold = limit then replace given solution by newer random scout solution
                     ELSE continue
                     Memorize best solution in iteration;
 END;
 UNTIL cycle = maxiter
```

Figure 2.10:- ABC Pseudo-Code

3 ARTIFICIAL BEE COLONY WITH NEIGHBOURHOOD OPERATOR

3.1 CONCEPT OF NEIGHBOURHOOD

For the successful execution of any population based stochastic search technique, regardless whether it is inspired by EA or SI, a proper trade-off must be achieved between the explorative and exploitative part of the search process. Exploitation is defined as the process by which the search process converges onto a global optimal solution amongst huge amounts of search space, that is exploitation is that aspect that helps the algorithm to quickly explore and search through huge volumes of search-space. However a search space fully inclined towards exploitation is not always desirable. Exploration is that aspect of the search operation that enables to explore through huge volumes of data search space and look for newer and innovative solutions. Both of them have their own set of advantages and disadvantages, exploitation is quicker and better in exploring global optimums, while exploration is better in local search and exploring newer options. The solution is to get an optimum compromise between the two aspects of the search. This is done by following a neighbourhood pathway of approach.

The neighbourhood [17][30] of any vector in a search space is defined as a set of other close-by parametric vectors that it is connected to; that is to speak the neighbourhood of the vector is area around it where it has connections to the other vectors nearby to it, that is it considers and deliberates their experiences before updating it's position. For local neighbourhood models, whenever there is a change in an individual population parameter, only the immediate neighbours that are directly connected are affected. Those neighbours that aren't in the immediate vicinity are not effected until the primary neighbours that are directly connected become highly viable themselves. This results in an delay in the information transfer throughout the breadth of the population regarding the position of the best solution in each such neighbourhood. Therefore there is a very good chance that the encompassed population doesn't get attracted to specific points that is get trapped in points of local minimas. Thereby placing a guarantee that the overall population search maintains a fine balance between the exploitative and the explorative parameters, and the best alternatives get selected. It now becomes necessary to define a new parameter that integrate both the global as well as the local aspects of the search inside the final solutions vectors. This is called as the weight vector, the weight vector is a variable that can be tuned so as to maintain the balance of exploitative of the explorative search aspects as per the requirements of the user.

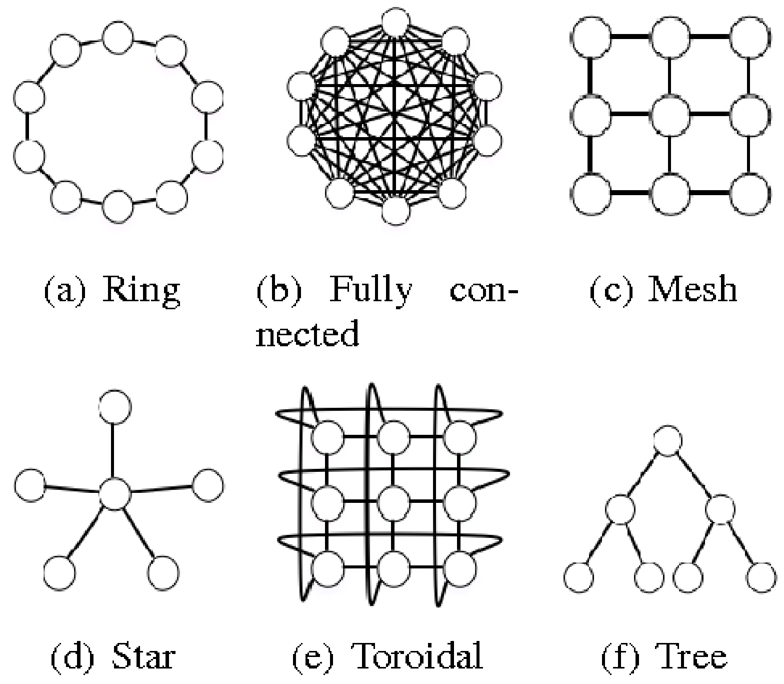


Figure 3.1:- Shows the various network topologies that may be used in a stochastic search space.

The above figure shows some of the commonly used topologies as per which neighbourhood structures may be designed, each of the above topologies have there own set of advantages and disadvantages. Some of the popular topologies[31][32] have been discussed in brief.

1. Ring Topology :- Ring Topology is an neighbourhood model structure in which every potential solution has exactly two neighbours to which it is connected and all information transfer takes through the immediate neighbour members. Ring topology. Typically, all messages travel through a ring in the same direction.
2. Mesh Topology :- It is an neighbourhood arrangement where the potential solutions are connected directly and non-hierarchically to as many other potential solutions as possible and mutual cooperation is achieved.

3. Star Topology :- It is a neighbourhood model in which in which all the solutions are individually connected to a central connection point, through which all information transfer takes place.

4. Tree topology :- It is a special type of neighbourhood structure in which all connected potential solutions are arranged like the branches of a tree.

5. A fully-connected network:-Also known as complete topology, it is a neighbourhood formation topology in which there is a direct link between all pairs of solutions and direct information transfer is possible.

For our purposes we prefer using the simplest ring topology structure

The details of which are mentioned in the next section :-

3.2 PROPOSED METHODOLOGY

3.2.1 Neighbourhood definations

Our approach is to employ the ABC Algorithm with a mutated neighbourhood operator. We start off by defining our problem, which is multidimensional or a multimodal optimisation problem. Let an arbitrary problem consisting of a D Dimensional parameter vector $j\ where\ j\ \in [1,2, \dots D]$ be initialized with a total population NP $= [x_{1j}, x_{2j}, \dots\dots x_{Nj}]$ where every individual D dimensional vector can be defined as $x_{ij}(\ where\ i = 1,2 \dots\dots NP)$. The vector indexes are randomly selected during initialization to ensure diversity of the solution. It has to be clarified right at the beginning that out the total population, the requisite population is obtained by splitting the given population into two equal populations of employed bees and onlooker bees. The number of food sources, that is potential solutions is equal to the number of employed bees. In retrospect, we must do a quick revision of the ABC algorithm. ABC Algorithm is basically a search based optimization that is modelled based on the food gathering behaviour of honey bees. In ABC the algorithm is built around the classification of the honey bees into three types, the employed bees, the onlooker bees and the scout bees. The employed and onlooker bees are tasked with finding the optimal solution based on the evaluation of the fitness parameters based on evaluation based on performance measuring benchmark functions, and then selecting based on probabilistic terms. Let the number of the employed bees be denoted by ENP while the onlooker bees are denoted by ONP, where $ENP = ONP$. The scout bees are tasked with the explorative aspect of the solution, that is finding newer and random food sources. However the number of scouts are very minimal, as scouts are only formed when a respective food source of a employed bees expires and it scouts out for newer solutions. This newer solution is determined against fitness evaluation.

It is this aspect of ABC that we wish to improve by employed a mutated neighbourhood operator. By selecting a global and local neighbourhood on every trail, we hope that the possibilities of both exploration and exploitation aspects of the search are maximized. Let us now come back to defining our neighbourhood parameters. Our neighbourhood will be organized with the ring topology Now for every corresponding x_{iG} vector we define a neighbourhood of radius y where $y \in [0\ to\ \frac{(ENP-1)}{2}]$.

This is done to ensure that the size of the neighbourhood lies within the size of the evaluated population, in order words :-

$2y + 1 \leq ENP$ where each vector falls in the range of $[x_{i-y\,j}, \dots . x_{ij}, \dots . x_{i+y\,j}]$. The arrangement of the vectors in the neighbourhood space is in the form of a ring where the vector $X_{i\,G}$ lies sandwiched between the vectors $X_{EN\,j}$ and $X_{i\,j}$. Now from these arrangement we select the best(fittest) vector solution based on fitness evaluation of the entire neighbourhood. As for the globally fit solution it is found at the initialization phase of a un-modified ABC algorithm by fitness evaluations. It is to be mentioned here that in the course of evaluation by the normal ABC algorithm the so formed global best solution is subjected to several corrections based on evaluation with respect to the fitness function.

Let us assume that the locally best and the globally best solutions obtained be represented as $v_{i\,j\,local\_best}$ and $v_{i\,j\,global\_best}$ respectively.

These two solutions are now effectively combined by a scalar weight factor w $where$ $w \in (0,1)$. The parameter w is used to fetch the effective solution $v_{i\,j}$ which maybe dominated by the global or the local vectors as per the values of w set by the user.

$$v_{i\,j} = w * \left(v_{i\,j\,global_{best}}\right) + (1 - w) * (v_{i\,j\,local_{best}}) \dots (3.1)$$

The scalar factor discussed here is a highly important parameter since it determines whether the solution so formed is leaning towards exploitative tendencies ($w \sim 1$) or explorative options ($w \sim 0$). Setting the value of w to 1 makes the resultant solution a purely exploitative solution. In such a case there is no difference between the normally used ABC algorithm and our modification. The aim of our modification is to increase the exploration abilities of the ABC algorithm and setting w close to 1 completely negates our purpose. On the other hand setting the value of w closer to 0 makes the generated solution purely explorative. Such a solution would be very likely to be stuck in a local minima and never converge on the global minima value. Ideally the value of w should be so selected as to ensure a healthy balance between exploitation and exploration

3.2.2 Defining the ABC approach

As discussed in section 2.32 of chapter 2, we start off the ABC algorithm by defining the number of food sources, the number of food sources being set to the number of employed bees. The number of onlooker bees are set equal to the number of employer bees. The limiting parameter so as to, define the attempts of numbers of retrials to exploit a food, following the expiry of which the food source is rejected, is also selected. This parameter is called as the 'limit' and it is a function of the dimension of the problem in hand and the number of onlooker bees. In our case we have tried to make the 'limit' factor dynamic by multiplying another weight factor in the range (0,1) . The third constraint of the ABC algorithm is the number of iterative cycles that the algorithm is allowed to execute before being terminated. In brief the optimization problem given to us is initialized and populated by the ENP number of possible solutions. If by any instance any food source gets exhausted the corresponding employed bees becomes scout bees. Scout bees are the explorers of the colony. Scout bees do not follow any specific guidance as *per se* but are motivated by internal motivations or react to external stimulus. Since scout bees are primarily concerned with exploring newer solutions, they are characterized by lower search cost values and a low solution quality average. However scouts can accidentally discover extremely rich and entirely unknown newer food sources. The limit parameter is the responsible for controlling the scout exploration. The search carried out by the modified ABC algorithm can be given as follows:-

- The employed bees take charge and food sources are identified and within the global and local neighbourhoods spaces respectively.
- The information regarding the food sources are now shared by the employer bees to the onlooker bees within the bee hive and the onlooker bees are fed with the choices of a wide variety of rich food sources.
- Onlooker bees select a food source by combining another search in the global and neighbourhood spaces, and the best food sources are selected based on a greedy selection process.
- By virtue of the limiting parameter when it so happens that when a food source that already had an employed bees foraging it, is exhausted and or it is abandoned in favour of better available food

sources, the employed bee turns into a scout and initiates a random search for a newer food source.

3.2.2.1 The initialization phase

The initial population consists of food sources where each food sources is symbolized by a D Dimensional vector represented as $j\, where\, j \in [1,2,3 \ldots .D]$, the resulting populated vector being represented as $x_{ij}\ where\ i \in [1,2,3 \ldots ENP]$ where ENP is the number of food sources, which is the number of employed bees present. The food sources are initialized by the following equation:-

$$x_{ij} = \ x_{\min j} + rand(0,1) * \left(x_{\max j} + \ x_{\min j}\right) \ldots (3.2)$$

3.2.2.2 Employed bee phase

This is the second phase of ABC algorithm. In this step the existing food sources are replaced with innovative newer food sources having better value of fitness calculated with the help of a fitness function. The local and global fitness are individually calculated by comparing it with an objective function. The resultant solutions are obtained after applying the scalar factor w on the respective solutions. This newer solutions are represented as :-

$$v_{i\,j} = w * \left(v_{i\,j\,global_{best}}\right) + (1 - w) * \left(v_{i\,j\,local_{best}}\right) \ldots (3.3)$$

$$v_{ij} = \ x_{ij} + rand(0,1) * (x_{ij} - x_{kj}) \ldots (3.4)$$

Where j ∈ {1,2….D} and where k ∈ {1,2….ENP} and k should be chosen that k≠i . This is done so as to ensure definitive improvements in the newer candidate solutions.

3.2.2.3 Onlooker Bees phase

The third phase of ABC algorithm is the onlooker bee phase. In this phase the information is transferred from the employer bees to the onlooker bees in form of fitness. Every food source is judged based on it’s evaluation based on fitness function. The fitness function is calculated on the basis of an objective function. Ultimately the highest quality solutions are selected based on the fitness function and selected based on probability which is calculated based on the fittest set of solutions. The concept of neighbourhood is applied again and the newer local and global solutions $v_{i\,j\,local\_best}$ and $v_{i\,j\,global\_best}$ are obtained.

The equivalent solution $v_{i\,j}$ is obtained. The obtained solution is again subjected to fitness evaluations. The advantages of subjecting the solutions generated by the employer bees to another rounds to fitness trails is to re-check for flaws and to iron out any shortcomings. Thus the onlooker bees are spoilt with the choices of the best solutions, based on fitness evaluation, available to them. The received solution is again modified by the onlooker bees and further checked for the fitness value.

Calculation of fitness function :-

Assumption :- Used objective function is greater than zero

$$Fitness\ function = \frac{1}{2*(objective\ function)+epsilon} \ldots(3.5)$$

where objective function is the performance evaluation function used and where 'epsilon' is a random uniform scalar value between (-1,1).

The probability selection is determined by selecting out those solutions which have the maximum fitness function amongst the total number of solutions available

$$P_{ij} = \frac{fit_i}{\sum_{i=1}^{NP} fit_i} \ldots(3.6)$$

3.2.2.4 Scout Bee Phase

The final phase is the scout phase where there is an attempt to foster newer random solutions. There is a threshold parameter that is defined at the start of the algorithm. The threshold parameter counts the number of trials that has taken place, and whenever and wherever the value of the threshold exceeds or equals the value of the limit, the scout phase is started and the threshold value is recalibrated to zero. The Newer food sources are generated as:-

$$x_{ij} = x_{\min j} + rand(0,1) * (x_{\max j} - x_{\min j}) \ldots 3.7$$

Since the scout phase is a purely explorative step, we donot apply any neighbourhood evaluations here. After the scout bee phase is completed, the algorithm conditions are evaluated and the optimized solutions are displayed after the algorithm terminates.

3.2.2.5 ABC with employed neighbourhood mutated pseudo code

```
BEGIN
    INITIALIZE :- Generate initial population;
    EVALUATE population fitness;
    ITER = 1;
    REPEAT
        FOR every employed bee
        generate new global solution;
        create neighbourhood and evaluate local solution;
        calculate local neighbourhood fitness;
        calculate global fitness;
        evaluate overall fitness expression and apply greedy selection process;
        calculate the probability values p;
        END;
        FOR every onlooker bee
        select best solution depending on p;
        generate new global solution;
        create neighbourhood and evaluate local solution;
        calculate local neighbourhood fitness;
        calculate global fitness;
        evaluate overall fitness expression and apply greedy selection process;
        END;
        IF threshold = limit then replace given solution by newer random scout solution
        ELSE continue
        Memorize best solution in iteration;
END;
UNTIL cycle = maxiter
```

Figure 3.2:- ABC neighbourhood with proposed mutated operator pseudo-code.

3.3 EXPERIMENTS

The evaluation of the performance of the ABC algorithm are done by some classical benchmark function given by Krink et al[2]. The results obtained by the proposed modified ABC have been compared with that of the classical ABC algorithm. The performances have been compared based on examination of the number of tried iterative cycles, the variation of the number of the initialized populated solutions, and also comparisions have been based on the radius of the defined neighbourhood in the region around a selected possible, solution. The percentages of the employed and onlooker bees have been kept exactly 50 to obtain the best possible results as suggested by D. Karaboga[27].We hoped that by using a local neighbourhood model in addition of the global solution, we would be able to obtain a better set of performances

Parameters. We have kept the dimension of the problem constant while making subtle adjustments to other variables to tweak the algorithm towards better performance. The performance metric functions that we have used for our trails are the spherical function and the rastrigin function[]. While the spherical function represents an unimodal evaluating parameter, the rastrigin function is a multimodal benchmark function which is quite difficult to minimize to due it having a large number of local minimas. This matters because our objective function is basically a minimizing function and we obtain the corresponding fitness function from the objective function. The evaluating criteria have been kept the same for both of the evaluating functions, first the number of cycles in varied, then the number of population vectors is tinkered with, and at last the weight factor is varied.

3.3.1 Used Benchmark Functions :-

3.3.1.1 The Spherical function:-

The spherical function [2][33] is a unimodal optimization evaluating performance metric function. It is amongst the most widely used benchmark function, preferred by researchers worldwide. It is a convex, continuous, unimodal, function defined over an n^{th} dimensional space. It is a separable and differential function with a global minima at ‘0’. The surface plot and contour lines are shown in the below figures

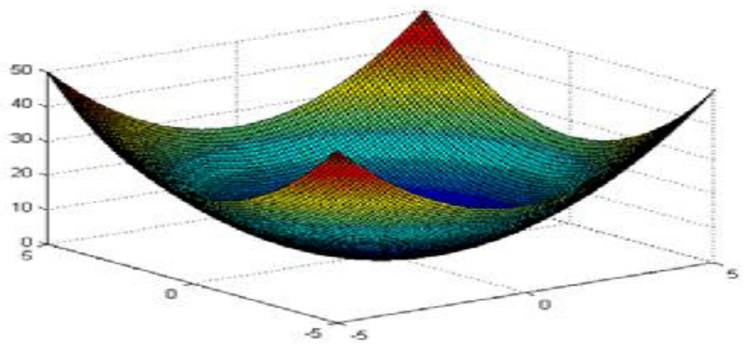


Figure 3.3:- Surface plot of a spherical function

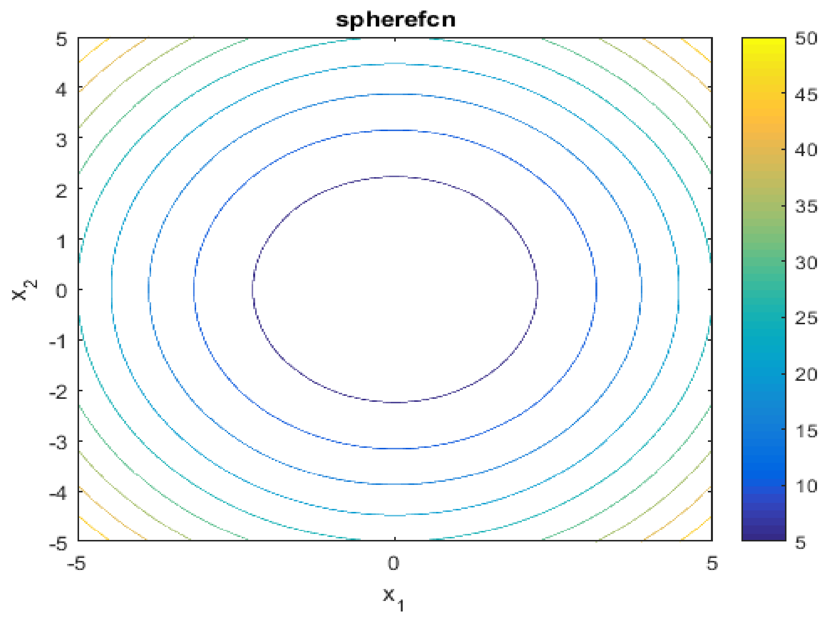


Figure 3.4:- Contour lines of a spherical function

The spherical function is defined as :-

$$f_1(x) = f(x_1, x_2, \ldots . x_n) = \sum_{i=1}^{n} x_i^2 \ldots (3.8)$$

For our test problem we used the Shifted Sphere function as a testing benchmark. The shifted sphere is defined as :-

$$f_1(x) = \sum_{i=1}^{n} x_i^2 + f_{bias} \ldots (3.9)$$

We set fbias to zero thus effectively reducing the function to

3.3.1.2 The Rastrigin Function:-

The Rastrigin function [2][33] is a modified application of the spherical function by adding of cosine modification to produce a multimodal function. It is a convex, continuous, multimodal, separable and differentiable function. It was first proposed by Rastrigin as a 2-dimensional function and has been generalized by Mühlenbein et al. It is defined in an n^{th} dimensional space. It is very difficult to find the optimal solution to this function due to the problem of it having many local minima, and the possibility of the potential solution being trapped into one of the local minimas. The global minima is also defined at $= 0 \; where \;\; f(x) = 0$. The maxima and minima of our problem solutions is set in [-5,5]. The surface plot and contour lines are given below

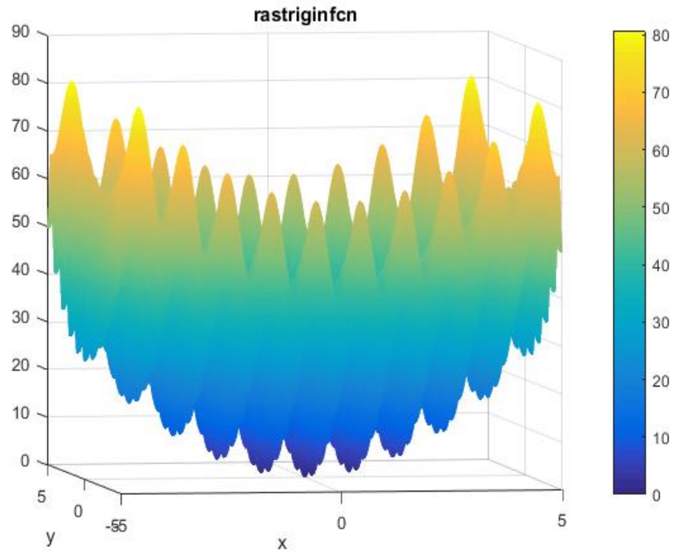


Figure 3.5:- Surface plot of Rastrigin function

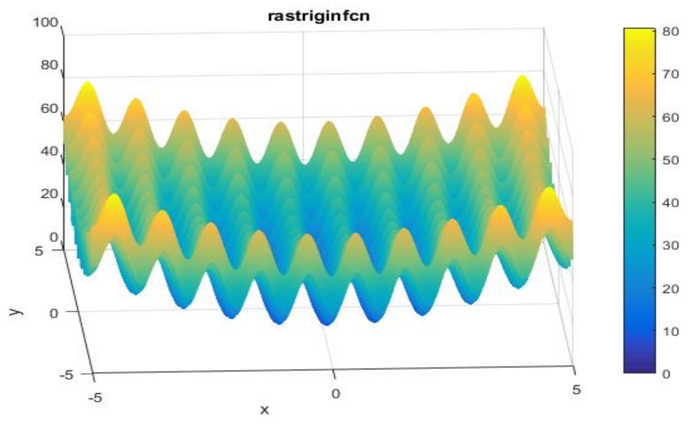


Figure 3.6:- Surface plot of Rastrigin function

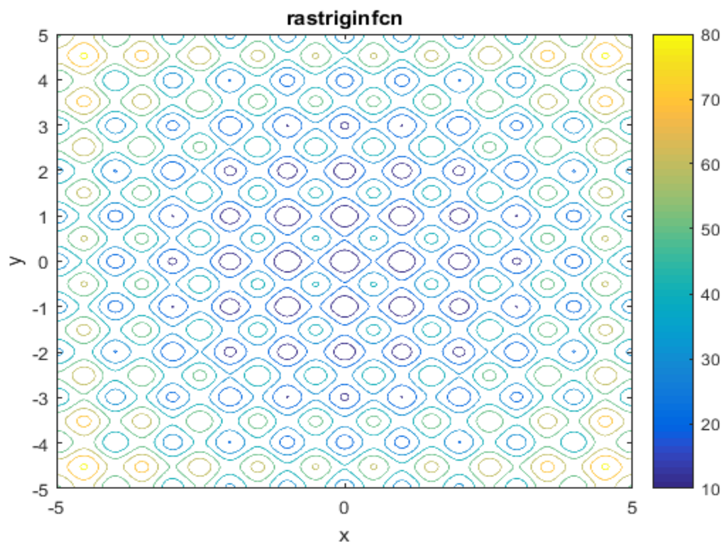


Figure 3.7:- Contour lines of Rastrigin

The Rastrigin function can be defined as :-

$$f_2(x) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10) \ldots (3.10)$$

Where i is varied from 1 to n, having a large of local minimas alongside a global minimums with the global minimum being at 0.

For actual purposes the shifted Rastrigin function is used, for simplicity purposes the f_{bias} is simply set to zero.

$$f_2(x) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10) + f_{bias} \ldots (3.11)$$

3.4 RESULTS AND COMPARATIVE ANALYSIS

Our multidimensional problem is subjected to trails numbers that will be varied and the effect on the progress of the problem will be observed. Furthermore the effect of variation of the population vectors will also be observed. The comparative analysis of the performance metric will be defined between conventional ABC and our modified version of ABC.

Our problem is a maximization problem and the quality of our solution is given by the value of the fitness function. Higher is the value of the fitness functions, better is the quality of our solutions. Our aim is to prove that by applying our mutated neighbourhood modified ABC, the fitness function parameters show an improved performance over a conventional ABC.

3.4.1 Comparative Analysis based variation of number of trials:-

We set our employed bee population to 20 represented here as ENP, the Dimension D is kept constant throughout our evaluation to 30, the limit is dynamically variable, the fitness function evaluated

3.4.1.1 Number of iterations set to 1000. Comparative analysis will be done between spherical and rastrigin functions:-

Spherical function evaluation					Rastrigin function evaluation			
Conventional ABC		Modified ABC			Conventional ABC	Modified ABC		
Maximum value of fitness	0.000018	0.000019	0.000019	0.000019	0.0020	0.0021	0.0026	0.0021
Mean value of fitness	0.000011	0.000013	0.000011	0.000011	0.0018	0.0019	0.0019	0.0019
Minimum value of fitness	0.000005	0.000011	0.000010	0.000007	0.0011	0.0017	0.0016	0.0011
Weight factor	1	0.5	0.5	0.5	1	0.5	0.5	0.5
Neighbourhood Radius	Not applicable	1	3	5	Not applicable	1	3	5
Limit	62	44	57	35	56	104	210	121
Employed bee population	20	20	20	20	20	20	20	20
Dimension	30	30	30	30	30	30	30	30

Where the weight factor is set to 1 for simulating conventional ABC algorithm, and by setting the weight factor when set to 0.5 gives equal preference to the global and local neighbourhood search space and simulates our modified ABC model. The significance of weight factor will be explained on the basis of a comparative analysis on the basis of value of the weight function. The values of the fitness function is given as it's maximum value, it's minimum value and the mean value.

3.4.1.2 Number of iterations set to 1500

Spherical function evaluation					Rastrigin function evaluation			
Conventional ABC		Modified ABC			Conventional ABC	Modified ABC		
Maximum value of fitness	0.000013	0.000013	0.000014	0.000014	0.0022	0.0021	0.0023	0.0023
Mean value of fitness	0.000011	0.000011	0.000012	0.000011	0.0018	0.0020	0.0019	0.0020
Minimum value of fitness	0.000009	0.000010	0.000011	0.000009	0.0015	0.0018	0.0018	0.0019
Weight factor	1	0.5	0.5	0.5	1	0.5	0.5	0.5
Neighbourhood Radius	Not Applicable	1	3	5	Not Applicable	1	3	5
Limit	171	203	205	293	20	260	251	317
Employed bee population	20	20	20	20	20	20	20	20
Dimension	30	30	30	30	30	30	30	30

3.4.1.3 Number of iterations set to 2000

Spherical function evaluation					Rastrigin function evaluation			
Conventional ABC		Modified ABC			Conventional ABC	Modified ABC		
Maximum value of fitness	0.000014	0.000014	0.000017	0.000016	0.0023	0.0023	0.0024	0.0025
Mean value of fitness	0.000011	0.000012	0.000013	0.000014	0.0019	0.0020	0.0022	0.0023
Minimum value of fitness	0.000009	0.000009	0.000010	0.000013	0.0018	0.0019	0.0021	0.0020
Weight factor	1	0.5	0.5	0.5	1	0.5	0.5	0.5
Neighbourhood Radius	Not Applicable	1	3	5	Not Applicable	1	3	5
Limit	243	213	373	183	327	271	203	199
Employed bee population	20	20	20	20	20	20	20	20
Dimension	30	30	30	30	30	30	30	30

3.4.2 Comparative analysis based on variation of population solutions:-

For this comparative approach, we set the number of trials to 2000, and vary the population and observe the behavioural changes. The dimensions are kept fixed at 30. The number of employed bees is varied and the results are observed for the sets of 20, 50 and 100 employed bees.

3.4.2.1 :- Iterations =2000, setting of employed bees population to 20.

Spherical function evaluation					Rastrigin function evaluation			
Conventional ABC		Modified ABC			Conventional ABC	Modified ABC		
Maximum value of fitness	0.000014	0.000014	0.000018	0.000017	0.0021	0.0022	0.0023	0.0024
Mean value of fitness	0.000011	0.000013	0.000011	0.000012	0.0019	0.0020	0.0020	0.0019
Minimum value of fitness	0.000009	0.000012	0.000011	0.000012	0.0018	0.0017	0.0019	0.0018
Weight factor	1	0.5	0.5	0.5	1	0.5	0.5	0.5
Neighbourhood Radius	Not Applicable	1	3	5	Not Applicable	1	3	5
Limit	197	108	59	223	145	230	373	247
Employed bee population	20	20	20	20	20	20	20	20
Dimension	30	30	30	30	30	30	30	30
Iterations	2000	2000	2000	2000	2000	2000	2000	2000

3.4.2.2 Iterations =2000, setting of employed bees population to 50

Spherical function evaluation					Rastrigin function evaluation			
Conventional ABC		Modified ABC			Conventional ABC	Modified ABC		
Maximum value of fitness	0.000014	0.000019	0.000016	0.000017	0.0020	0.0021	0.0022	0.0023
Mean value of fitness	0.000011	0.000010	0.000010	0.000011	0.0019	0.0019	0.0019	0.0019
Minimum value of fitness	0.000010	0.000007	0.000009	0.000008	0.0017	0.0018	0.0018	0.0017
Weight factor	1	0.5	0.5	0.5	1	0.5	0.5	0.5
Neighbourhood Radius	Not Applicable	1	3	5	Not Applicable	1	3	5
Limit	549	487	473	191	197	664	675	891
Employed bee population	50	50	50	50	50	50	50	50
Dimension	30	30	30	30	30	30	30	30
Iterations	2000	2000	2000	2000	2000	2000	2000	2000

3.4.2.3 Iteration =2000, setting of employed bees population to 100

Spherical function evaluation					Rastrigin function evaluation			
Conventional ABC		Modified ABC			Conventional ABC	Modified ABC		
Maximum value of fitness	0.000014	0.000015	0.000015	0.000016	0.0024	0.0021	0.0024	0.0026
Mean value of fitness	0.000010	0.000010	0.000010	0.000011	0.0019	0.0019	0.0018	0.0022
Minimum value of fitness	0.0000037	0.0000049	0.0000062	0.000010	0.0015	0.0015	0.0017	0.0015
Weight factor	1	0.5	0.5	0.5	1	0.5	0.5	0.5
Neighbourhood Radius	Not Applicable	1	3	5	Not Applicable	1	3	5
Limit	252	342	1167	61	1156	772	1327	1091
Employed bee population	100	100	100	100	100	100	100	100
Dimension	30	30	30	30	30	30	30	30
Iterations	2000	2000	2000	2000	2000	2000	2000	2000

3.4.3 Effect of variation of weight factor on the performance.

The weight factor is a crucial element in deciding the exploitative and explorative aspect of the search algorithm. Here we will evaluate the performance of our algorithm as the variation of the algorithm is evaluated from a purely explorative to a purely exploitative perception.

This analysis is done with only Rastrigin function, since proving the effect of the variation of weight factor is our primary objective. To ensure that we are able to prove the effect of the variation of the weight factor on our search algorithm we also keep the modified ABC with a constant defined neighbourhood radius of 3.

Effect of variation of the weight function by evaluating with Rastrigin benchmark function for ENP = 20 and number of iterations set to 2000.						
ABC with neighbourhood radius set to 3	w = 0.0	w = 0.2	w = 0.4	w = 0.6	w = 0.8	w = 1.0
Limit set	228	137	30	292	110	212
Maximum value of fitness function	0.0023	0.0024	0.0027	0.0024	0.0021	0.00207
Mean value of fitness functions	0.0017	0.0017	0.0017	0.0020	0.0019	0.0019
Minimum value of fitness functions	0.0012	0.0012	0.0011	0.0019	0.0018	0.0018

From our observations it is clear that the best performance of the ABC algorithm is achieved whenever the weight factor takes upon a value between 0.4 to 0.6, thus reaffirming our hypothesis that the performance of the ABC is improved whenever we are able to obtain a healthy balance between the global and local searching abilities. It was for these reasons that we kept the value of the weight factor as 0.5 for our performance analysis, based on the variation of parameters like the initialized population and number of iterative trials. By keeping the value of the weight factor at 0.5, the final generated solution used by us contained an equal influence of both the explorative and exploitative searches. Thus we are nearly always able to observe that our suggested modified algorithm always works better and gives better fitness evaluations against, both the of the evaluating benchmark functions.

As an example, we are presenting some screenshots of our simulation of our work, notice how the fitness function reached is always higher whenever, the number of population initialized is higher, and more importantly for the purpose of these thesis, the positive effect that introduction of a local neighbourhood has on boosting the better fitness of the obtained solutions.

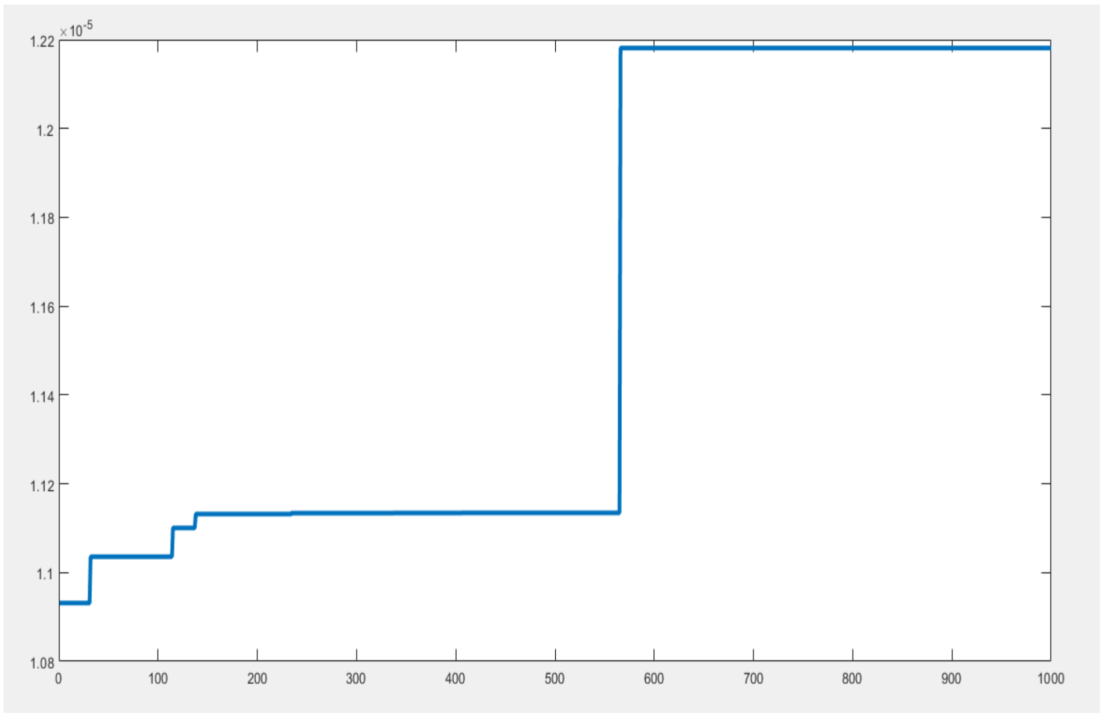


Figure 3.8:-A random simulation result of a initialized population of 20 with a local neighbourhood.

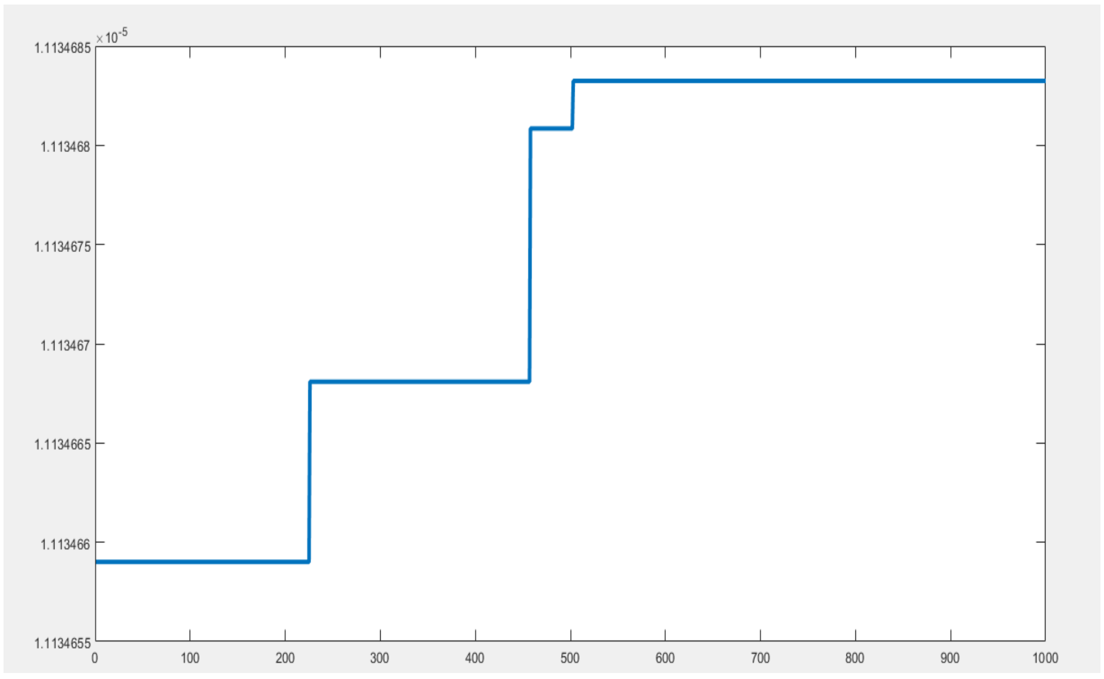


Figure 3.9:- The same population of 20 population vectors, when run without any mutated local neighbourhood.

Note that the above two figures are for figurative representation only. These two figures are the representation of two completely different simulation iterations and different iterations are very likely to give different results. However through our experimentation we are sufficiently confident in saying that through the use of the mutated neighbourhood vector, the quality of the obtained solutions generally shows an improvement, and the figures attached above are certainly indicative of the most likely obtained visual outcome representation, of two random runs, one running with and the other without a mutated neighbourhood vector.

3.5 CONCLUSION

Based on our experimental trials we were successful in proving introducing a local neighbourhood based mutation vector improves the performance for given function optimization for the ABC algorithm. For obtaining the best optimization performance out of the algorithm it is recommended to keep the weight factor of the algorithm between 0.4 and 0.6 to keep a healthy balance between the globally and locally generate fitness evaluations.

We observed that increasing the population size practically has little to no effect on the performance metrics of our algorithm, however if we go on increasing the population size gradually a limit is attained after which the performance of the ABC undergoes a sharp deterioration. Increasing the number of iterations has little to no effect on the increase in optimization efficiency after a certain value of iterations is reached and further evaluation is a wastage of computational resources.

The number of onlooker bee population and employer bee population is kept equal in an ABC based on recommendations of D. Karaboga, to make the algorithm perform at it’s highest efficiency while also minimizing the number of control parameters for the modified ABC.

4 CONCLUSION AND FUTURE DIRECTION

Since it’s proposal , The Artificial Bee Colony algorithm has become extremely popular due to it’s simple and robust nature, while having less number of control parameters. It’s effectiveness against complex and multi dimensional non-linear optimization problems have made it a highly popular choice of researchers. It has been applied in almost every applicable practical field including engineering, science and management where ABC algorithm is preferred over other competitive optimization algorithms. ABC has also been a highly popular area of research with various modifications and changes constantly being made upon it. In today’s era of science and application where accuracy and efficiency are of paramount importance while handling complex real world problems, the demands for robust, fast and accurate optimizers are an all time high. The ABC algorithm is one of the simplest and most effective swarm intelligence based algorithm offering good accuracy results for a variety of practical problems with different levels of complexity. ABC effectiveness lies in it’s versatility and applicability to a wide variety of problems. Alongside it’s compatriot the Particle swarm optimization (PSO), ABC is amongst the most widely used optimizers in the industry today, being applied to a plethora of real life problems. However ABC handles problems of higher dimensionality much better than the conventional PSO while the PSO handles much higher values of population vectors more seamlessly. ABC also suffers from it’s own share of shortcomings such as premature convergence or stagnation leading to loss of balance between intensification and diversification capabilities. The primary objective of this thesis has been to look out for improving the performance of the conventional ABC by using a neighbourhood mutated operator. The performance of the modified ABC has been compared with that of the conventional ABC for a multi-dimensional by evaluating with respect to uni-modal and multi-modal performance metric evaluation functions. The simulation results show that the modified ABC algorithm performs better than the conventional ABC algorithm and can be therefore efficiently employed to solve the multimodal engineering problems at hand, with high dimensionality.

There are a lot of scope for future exploration for the study conducted by these thesis. We employed a fixed neighbourhood radius for our evaluation, the option of a dynamic neighbourhood vector could be explored.

REFERENCES

[1] Evans.J (1992). Optimization algorithms for networks and graphs. CRC Press 2nd edition.

[2] Momin Jamil and Xin-She Yang, A literature survey of benchmark functions for global optimization problems, Int. Journal of Mathematical Modelling and Numerical Optimisation.

[3] Dorfman, Robert (1969). "An Economic Interpretation of Optimal Control Theory"

[4] Gregory, T.R. Evo Edu Outreach (2009) 2: 156

[5] I. Fister, X. S. Yang, J. Brest, and D. Fister, "A Brief Review of Nature-Inspired Algorithms for Optimization", Elektrotehniˇski Vestnik, Jul, 2013.

[6] Feng Weiwei, "Multi-objective Optimization Algorithm for Multimedia English Teaching (MOAMET) Based on Computer Network Traffic Prediction Model".

[7] P. Rakhshit, A. Chowdhury, A. Konar, "Differential Evolution Induced Many Objective Many Objective Optimization."

[8] H.K Singh, A. Isaacs, and T. Ray," A Pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems."

[9] J. J. Liang , C. T. Yue , B. Y. Qu, "Multimodal multi-objective optimization: A preliminary study"

[10] Richard Bellman. "The Theory of Dynamic Programming."

[11] Robert J. Richards," Darwin's Theory of Natural Selection and Its Moral Purpose ."

[12] A.E. Eiben, J.E. Smith, " Introduction to Evolutionary Computation."

[13] Felix Streichert, " Introduction to Evolutionary Algorithms."

[14] DE. Goldberg, "Genetic algorithms in search, optimization and machine learning".

[15] A. Parashar and K. K. Swankar, "Genetic algorithm using to the solution of unit commitment."

[16] David E. Goldberg, John H. Holland," Genetic Algorithms and Machine Learning."

[17] Das, Swagatam & Suganthan, Ponnuthurai. (2011). Differential Evolution: A Survey of the State-of-the-Art.. IEEE Trans. Evolutionary Computation. 15. 4-31.

[18] Das, Swagatam & Subhra Mullick, Sankha & Suganthan, Ponnuthurai. (2016). Recent Advances in Differential Evolution – An Updated Survey. Swarm and Evolutionary Computation. 27. 10.1016/j.swevo.2016.01.004.

[19] Huang Zhehuang & Chen Yidong. "An Improved Differential Evolution Algorithm Based on Adaptive Parameter."

[20] Dimitri P. Bertsekas, "Nonlinear Programming."

[21] Kenneth E. Kinnear, “Advances In Genetic Programming.”

[22] Shahrul Badariah Mat Sah, Vic Ciesielski, Daryl D’Souza, Marsha Berry,” Comparison between Genetic Algorithm and Genetic Programming Performance for Photomosaic Generation

[23] Ahmed, Hazem & Glasgow, Janice. (2012). Swarm Intelligence: Concepts, Models and Applications. 10.13140/2.1.1320.2568.

[24] Eric Bonabeau, Marco Dorigo and Guy Theraulaz**. “**Swarm Intelligence: From Natural to Artificial Systems.”

[25] Eric Bonabeau, “Swarm Intelligence.”

[26] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm”

[27] J. Kennedy and R. Eberhart, “Particle swarm optimization”, IEEE international conference on neural networks.

[28] M. Dorigo, V. Maniezzo and A. Colorni, “Ant system: optimization by a colony of cooperating agents.”

[29] D. Karaboga and B. Basturk, “On the performance of artificial bee colony (ABC) algorithm.

[30] Hui Wang, Zhijian Wu, Shahryar Rahnamayan, “ Differential Evolution Enhanced by Neighborhood Search.

[31] Lim, Francis. (2016). A Review-Analysis of Network Topologies for Microenterprises. 175-180. 10.14257/astl.2

[32] Janez Brest, Saso Greiner, Borko Bo ˇ skovic ˇ ´, Marjan Mernik, and Viljem Zumer ˇ , “Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems.”

”

APPENDIX 1

Matlab Code used in the algorithm :-

```
clc;
clear all;
close all;
global initial_flag;
initial_flag = 0;
% Defining of x variation parameters
%-----------------------------------------------------------------
x_min = [-100,-100,-100,-100,-100,-100,-600,-32,-5,-5,-0.5,-pi,-3,-100,-5,-5,-5,-5,-
5,-5,-5,-5,-5,-5,-5];
x_max = [100,100,100,100,100,100,600,32,5,5,0.5,pi,1,100,5,5,5,5,5,5,5,5,5,5,5];
func_num = 9;
% Parameter defination
%-------------------------------------------------------------------
maxiter = 2000;
D = 30;
enp = 100;
onp = 100;
y = 3;
limit = floor((rand*onp*D)/2);
e = unifrnd(-1,+1,+1);
w = 0.5;
% Randomize solution
%-------------------------------------------------------------------
for i = 1:enp
    for j = 1:D
        x(i,j) = x_min(func_num) + rand*(x_max(func_num)-x_min(func_num));
    end;
    f(i) = benchmark_func(x(i,:),func_num);
    fitt(i) = 1/(f(i)+e);
end;
[maxfit, index] = max(fitt);
x_globalbest(i,:) = x(index,:);
abandon(i) = 0;
% Iteration
%------------------------------------------------------------------
% Employed bees phase
for iter = 1:maxiter
    for i = 1:enp
        % Defining k
        k = floor(rand(1)*enp)+1;
        while k==i
            k = floor(rand(1)*enp)+1;
        end;
        % Defining j
        j = floor(rand(1)*D)+1;
        %------------------------------------------------------------------
        % Defining global variables
        vglobal(i,:) = x(i,:);
        % Formation of random global solution by employed bees
        vglobal(i,j) = x(i,j) + rand*(x(i,j)-x(k,j));
        % Verifying solution within limits
        if vglobal(i,j) < x_min(func_num)
            vglobal(i,j) = x_min(func_num);
        end;
        if vglobal(i,j) > x_max(func_num)
            vglobal(i,j) = x_max(func_num);
        end;

        %------------------------------------------------------------------
        % Defining local variables
        bnf = f(i);
        neighbour = [];
        for j = (i-y):(i+y)
            if j <= 0
                j = j+enp;
            end;
            if j>enp
                j = mod(j,enp);
            end;
            neighbour = [j,neighbour];
            if f(j) <= bnf
                bnf = f(j);
                x(i,:) = x(j,:);
            end;
        end;
        % Defining local solutions l
        l = floor(rand(1)*((2*y)+1))+1;
        while l==i
            l = floor(rand(1)*((2*y)+1))+1;
        end;
        j = floor(rand(1)*D)+1;
        vlocal(i,:) = x(i,:);
        % Formation of local random solution by employed bees
        vlocal(i,j) = x(i,j) + rand*(x(i,j) - x(l,j));
        % Verifying solution is within limits
        if vlocal(i,j) < x_min(func_num)
            vlocal(i,j) = x_min(func_num);
        end;
        if vlocal(i,j) > x_max(func_num)
            vlocal(i,j) = x_max(func_num);
        end;
        % Formation of combined solution
        for j = 1:D
            v(i,j) = (w)*(vglobal(i,j)) + (1-w)*(vlocal(i,j));
            if v(i,j) < x_min(func_num)
                v(i,j) = x_min(func_num);
            end;
            if v(i,j) > x_max(func_num)
                v(i,j) = x_max(func_num);
            end;
        end;
        %-----------------------------------------------------------------
        % Evaluating new fitness
        fie(i) = benchmark_func(v(i,j),func_num);
        fite(i) = 1/(fie(i)+e);
        %_________________________________________________________________
        % Check
        if fite(i) > fitt(i)
            abandon(i) = 0;
            x(i,:) = v(i,:);
            fitt(i) = fite(i);
            if fite(i) > maxfit
                x_globalbest(i,:) = v(i,:);
                maxfit = fite(i);
            end;
        else
            abandon(i) = abandon(i) + 1;
        end;
    end;
    %----------------------------------------------------------------------
    % Probability admission
    F = sum(fitt);
    for i = 1:enp
        p(i) = fitt(i)/F;
    end;
    %-----------------------------------------------------------------------
    % Onlooker bees
    i = 1;
    for trial = 1:onp
        if rand < p(i)
            % Global solution k and j creation
            k = floor(rand(1)*onp)+1;
            while k==i
                k = floor(rand(1)*onp)+1;
            end;
            j = floor(rand(1)*D)+1;
            vglobal(i,:) = x(i,:);
            % Formation of random global solution by employed bees
            vglobal(i,j) = x(i,j) + rand*(x(i,j)-x(k,j));
            % Verifying solution within limits
            if vglobal(i,j) < x_min(func_num)
                vglobal(i,j) = x_min(func_num);
            end;
            if vglobal(i,j) > x_max(func_num)
                vglobal(i,j) = x_max(func_num);
            end;
            %-----------------------------------------------------------------
            % Defining local variables
            bnf = f(i);
            neighbour = [];
            for j = (i-y):(i+y)
                if j <= 0
                    j = j+onp;
                end;
                if j>onp
                    j = mod(j,onp);
                end;
                neighbour = [j,neighbour];
                if f(j) < bnf
                    bnf = f(j);
                    x(i,:) = x(j,:);
                end;
            end;
            % Defining local solutions l
            l = floor(rand(1)*((2*y)+1))+1;
            while l==i
                l = floor(rand(1)*((2*y)+1))+1;
            end;
            j = floor(rand(1)*D)+1;
            vlocal(i,:) = x(i,:);
            % Formation of local random solution by employed bees
            vlocal(i,j) = x(i,j) + rand*(x(i,j) - x(l,j));
            % Verifying solution is within limits
            if vlocal(i,j) < x_min(func_num)
                vlocal(i,j) = x_min(func_num);
            end;
            if vlocal(i,j) > x_max(func_num)
                vlocal(i,j) = x_max(func_num);
            end;
            % Formation of combined solution
            for j = 1:D
                v(i,j) = (w)*(vglobal(i,j)) + (1-w)*(vlocal(i,j));
                if v(i,j) < x_min(func_num)
                    v(i,j) = x_min(func_num);
                end;
                if v(i,j) > x_max(func_num)
                    v(i,j) = x_max(func_num);
                end;
            end;
            %----------------------------------------------------------------
            % Evaluating new fitness
            fio(i) = benchmark_func(v(i,j),func_num);
            fito(i) = 1/(fio(i)+e);
            %________________________________________________________________
            % Check
            if fito(i) > fitt(i)
                abandon(i) = 0;
                x(i,:) = v(i,:);
                fitt(i) = fito(i);
                if fito(i) > maxfit
                    x_globalbest(i,:) = v(i,:);
                    maxfit = fito(i);
                end;
            else
                abandon(i) = abandon(i)+1;
            end;
        else
            trial = trial -1;
        end;
        i = mod(i,onp);
    end;
    %--------------------------------------------------------------------------
    % Scout phase
    [maxabandon,in] = max(abandon);
    if abandon(in) >= limit
        for j = 1:D
            x(in,j) = x_min(func_num) + rand*(x_max(func_num) - x_min(func_num));
        end;
        fsc(i) = benchmark_func(x(in,:),func_num);
        fitsc(i) = 1/(fsc(i)+e);
        abandon(i) = 0;
        % Verification
        if fitsc(i) > maxfit
            x_globalbest(i,:) = x(in,:);
            maxfit = fitsc(i);
        end;
    end;
    global_best_soln = max(x_globalbest);

    disp(['Iteration = ' num2str(iter) '==> maxfit = ' num2str(maxfit)]);
    z(iter) = maxfit;

    plot(z,'LineWidth',3);





end;
```