Accelerating Ant Colony Optimization By Using Local Search

Nabila Tabassum ID: 11301006

Maruful Haque ID:11201003

A DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering

Supervisor: Dilruba Showkat

August 2015

Contact Information:

Author 01:
Nabila Tabassum
E-mail: nabila.tabassum1991@gmail.com

Author 02:
MarufulHaque
E-mail: shourov.sheikh@gmail.com

Supervisor:
DilrubaShowkat
E-mail: dilrubashowkat@gmail.com
Lecturer
Department of Computer Science and Engineering
Brac University
Dhaka

Table of Contents

CHAPTER 3

CHAPTER 4

CHAPTER 5

CHAPTER 6

CHAPTER 7

LIST OF FLOW CHART :

LIST OF FIGURE:

LIST OF TABLE:

Acknowledgement

First and foremost, we would like to express our gratitude to Almighty Allah (SWT) who gave me the opportunity, determination, strength and intelligence to complete my thesis. We want to acknowledge our fellow classmates who have consistently support us throughout the thesis.We would like to thank our supervisor Dilruba Showkat sincerely for her consistent supervision, guidance and unflinching encouragement in accomplishing our work and help us to present our idea properly.

Abstract

Optimization is very important fact in terms of taking decision in mathematics, statistics, computer science and real life problem solving or decision making application. Many different optimization techniques have been developed for solving such functional problem. In order to solving various problem computer Science introduce evolutionary optimization algorithm and their hybrid. In recent years, test functions are using to validate new optimization algorithms and to compare the performance with other existing algorithm. There are many Single Object Optimization algorithm proposed earlier. For example: ACO, PSO, ABC. ACO is a popular optimization technique for solving hard combination mathematical optimization problem. In this paper, we run ACO upon five benchmark function and modified the parameter of ACO in order to perform SBX crossover and polynomial mutation. The proposed algorithm SBXACO is tested upon some benchmark function under both static and dynamic to evaluate performances. We choose wide range of benchmark function and compare results with existing DE and its hybrid DEahcSPX from other literature are also presented here.

CHAPTER 1

Introduction

Science, invention, applied mathematics, economic analysis, industrial, technological and managerial decisions making need solution in an optimal way. In this modern era, day by day the world becomes more first at the same time more and more complex and competitive so that decision making must be taken in an optimal way for better results in a faster way. Therefore optimization is very important and concerning act of obtaining the best result under given situations. This is the reason behind why optimization has been a popular research topic for decades. Optimization originated in the 1940s, when the British military faced the problem of allocating limited resources (for example fighter airplanes, submarines and so on) to several activities [6]. In computer science over the decades, several researchers have generated different solutions to linear and non-liner optimization problems. Among them evolutionary algorithms is most popular and interesting part of modern computer science. Evolutionary algorithm are biology-inspired soft computing methods have been widely used in different optimization problem solving cases. Mathematically an optimization problem has a fitness function, describing the problem under a set of constraints which represents the solution space for the problem [1]. Especially, considering dynamic optimization problems it is quite interesting how their objective functions changed over time, which causes changes in the position of optima as well as the characteristics of the search space. This leads to the fact where existing optima may disappear, while new optima may appear. Optimization under the dynamic environments is a challenging task that attracts great attention [8].However, most of the traditional optimization techniques have calculated the first derivatives to locate the optima on a given constrained surface but modern optimization methods which are known as nontraditional optimization works dynamically for constrained and unconstrained problems. Due to the difficulties in evaluation the first derivative for many rough and discontinuous optimization spaces, several derivatives free optimization methods have been constructed in recent time [15]. There are many single optimization algorithm have been introduced in last few years. But there is no known single optimization method available for solving all kind of optimization problems. In order to develop the efficiency a lot of hybrid single optimization problem have been developed for solving different types of optimization problems more efficiently. It becomes very popular and useful in terms of research now days. The modern single optimization methods (sometimes called nontraditional optimization methods) are very powerful. These are particle swarm optimization algorithm, neural networks, genetic algorithms, ant colony optimization, differential evolution, artificial immune systems, and fuzzy optimization, the Clonal Selection Algorithm (CSA), an important branch of the Artificial Immune Systems (AIS) [6] [7]. The Ant Colony Optimization (ACO) algorithm is another emerging approach mimicking the foraging behavior of the ant species [8].The complex social behaviors of ants have been much studied by science, and computer scientists are now finding that these behavior patterns can provide models for solving

difficult combinatorial optimization problems [3].ACO is the member of swarm intelligence family and it can constitutes some metaheuristic optimizations. The novel algorithms inspired by aspect of ant behavior, the ability to find what computer scientists would call shortest paths, has become the field of ant colony optimization (ACO), the most successful and widely recognized algorithmic technique based on ant behavior.ACO is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graph. Thisalgorithm is initially proposed by Marco Dorigo in 1992 in his PHD thesis [2]. In his thesis work he presents an overview of this rapidly growing field, from its theoretical inception to practical applications, including descriptions of many available ACO algorithms and their uses. In his book ACO is not only introduced but also surveys on ACO applications now in use, including routing, assignment, scheduling, subset, machine learning, and bioinformatics problems. AntNet, an ACO algorithm designed for the network routing problem, is described in detail. On the other hand in EA cross over mutation is very popularmethod of GA. SBX is used as one point cross over properties for binary GA. It was proposed in 1995 by Deb and Agrawal. Polynomial is a variant mutation operator in GA. These two are real genetic operation where fitness function and constrained function are calculated as normal. Researchers use this operator to propose any hybrid optimization algorithm for variation and efficient results. However, these powerful optimization methods (ACO) have their inherent shortcomings and limitations. As we know, fusion and hybrid optimization can give efficient result in some previous case and it increases efficiency level. Therefore, in this paper, a hybrid algorithm has been introduced using SBX as crossover operator and Polynomial mutation is performed for mutation. We compare basic ACO with our hybrid SBXACO by test them upon five benchmark functions.

1.1 ACO is member of swarm intelligence

Swarm intelligence is a kind of discipline that deals with the study of self-organizing processes both in nature and in artificial systems. Recently, computer scientists introduced algorithms inspired by these models in order to solve difficult various complex computational problems.ACO is successfully included in swarm intelligence family where the concept is adopted from folk of ants behaviors. The mathematical function of ACO is introduced as such concept. The basic ACO is single object Optimization but very recently MACO is also introduced.

1.2 Motivation

Aco is proposed on 1992, since then it has been use to solve various problems as like as TSP, bioinformatics, networking, machine learning, variant problem solving and many other applications. But an ACO method does not always work well and still has room for improvement on some benchmark functions. This thesis discusses conceptual ACO algorithm and its modifications. It also describes different types of ACO algorithms and flowcharts recent works, advanced topic, and application areas of ACO.

1.3 Research Questions

This thesis aims to answer following questions:

Q.1: How functions of ACO parameterized?

Q.2: How efficiently ACO works while tested upon bench mark or test functions?

Q.3: Can we use SBX cross over and Polynomial mutation with ACO?How?

Q.4: How efficiently SBXACO works while tested upon benchmark function?

Q.5: Which one gives best solution? Compare efficiency?

CHAPTER 2

Background:

This chapter reviews some of the basic definitions related to this thesis.

2.1 Optimization:

The aim of optimization is to determine the best-suited solution of a problem undera given set of constraints. For example, in mathematical problem there are several equation where if wide range taken it become massive problem to choose the best one. Even in industries and scientific experiment optimization is needed to choose the best way possible. Optimization refer to both minimization and maximization tasks. Since the maximization of any function is mathematically equivalent to the minimization of its additive inverse, the term minimization and optimization are used inter changeably [6]. Optimization problems may be linear or nonlinear.

2.1.1 Constrained Optimization

Many optimization problems require that some of the decision variables satisfy certain limitations, for instance, all the variables must be non-negative. Such typesof problems are said to be *constrained optimization problems* [4] [8] [11] and defined as,
Minimize *f(x)*, $x=(x_1,x_2,x_3\ldots\ x_n)$
Subject to $g_m(x)<=0.M=1,\ 2\ldots\ldots n_g$
$h_m(x)=0,m=ng+1,\ldots\ldots,n_g+n_h$
$\forall\ x\in R^n$
Where n_g and n_h are the number of inequality constraints respectively.

2.1.2 Unconstrained Optimization

Many optimization problems place no restrictions on the values of that can be assigned to variables of the problem. The feasible space is simply the whole search space. Such types of problems are said to be *unconstrained optimization problems* [4] and defined asminimize
$f(x),\ x\in R^n$. Where n is the dimension of x.

2.1.3 Dynamic Optimization

Many optimization problems have objective functions that change over time and such changes in objective function cause changes in the position of optima. These types of problems are said to be *dynamic optimization problems* [4].

2.2.1 Single Object Optimization:

Many real-world decision making problems need to achieve several objectives: minimize risks, maximize reliability, minimize deviations from desired levels, minimize cost, etc. The main goal of single-objective (SO) optimization is to find the “best” solution, which corresponds to the minimum or maximum value of a single objective function that lumps all different objectives into one. This type of optimization is useful as a tool which should pro-vide decision makers with insights into the nature of the problem, but usually cannot provide a set of alternative solutions that trade different objectives against each other.

ABC (Artificial Bee Colony):

The ABC algorithm is proposed by Karaboga [16] in 2005 and the performance of ABC is analyzed in 2007 [15].The ABC algorithm is developed by inspecting the behaviors of the real bees on findingfood source, which is called the nectar, and sharing the information of food sources to the bees in the nest. In the ABC, the artificial agents are defined and classified into three types, namely, the employed bee, the onlooker bee, and the scout. Each of them plays different role in the process: the employed bee stays on a food source and provides the neighborhood of the source in its memory; the onlooker gets the information of food sources from the employed bees in the hive and select one of the food source to gather thenectar; and the scout is responsible for finding new food, the new nectar, sources[14].

Flow chart of Basic ABC:

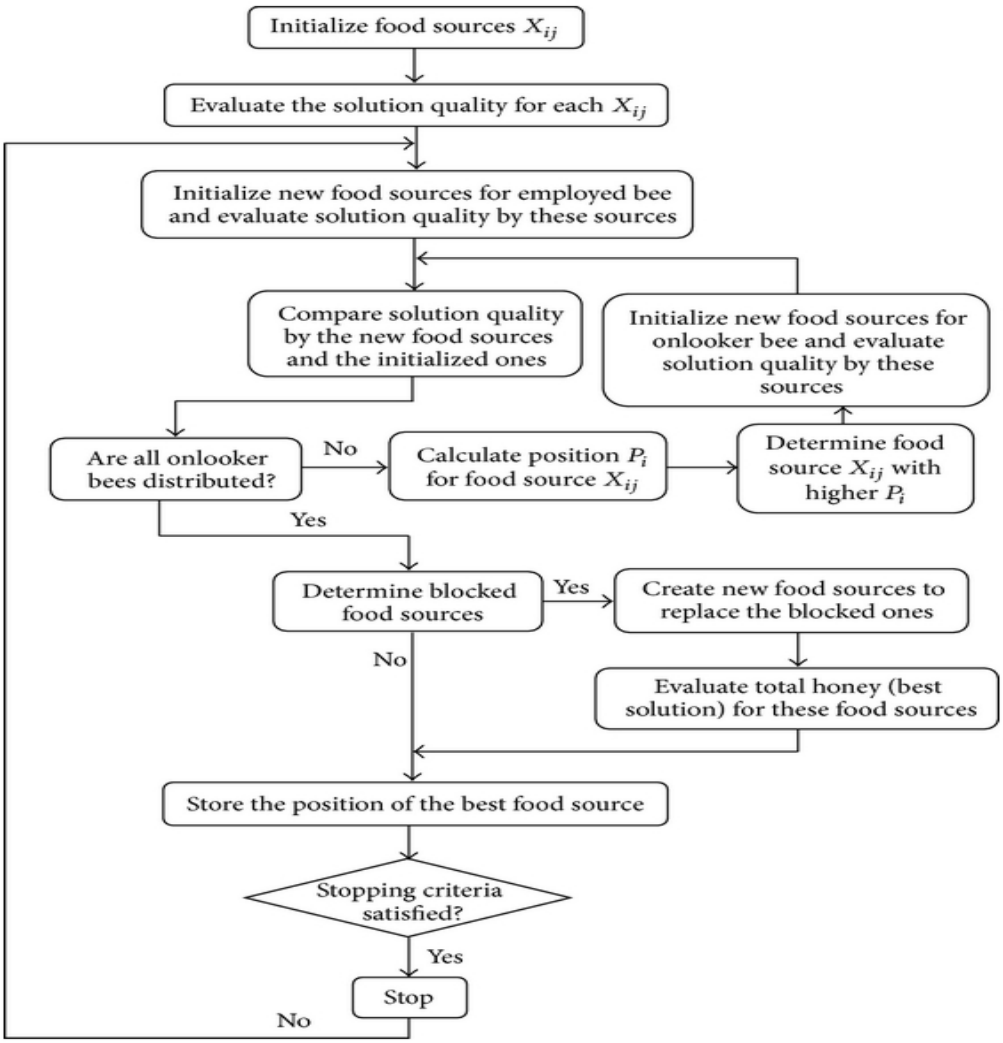

Figure 2.1:*flow chart of ABC*

PSO(Particle Swarm Optimization):

The Particle Swarm Optimization algorithm (abbreviated as PSO) is a novel population-based stochastic search algorithm and an alternative solution to the complex non-linear optimization problem. The PSO algorithm was first introduced by Dr. Kennedy and Dr. Eberhart in 1995 and its basic idea was originally inspired by simulation of the social behavior of animals such as bird flocking, fish schooling and so on [10].In PSO, each member of the population is called a particle and the population is called a swarm. Starting with a randomly initialized population and moving in randomly chosen directions, each particle goes through the searching space and remembers the best previous positions of itself and its neighbors. Particles of a swarm communicate good positions to each other as well as dynamically adjust their own position and velocity derived from the best position of all particles. The next step begins when all particles have been moved. Finally, all particles tend to fly towards better and better positions over the searching process until the swarm move to close to an optimum of the fitness function.

$f: R^n \rightarrow R.$

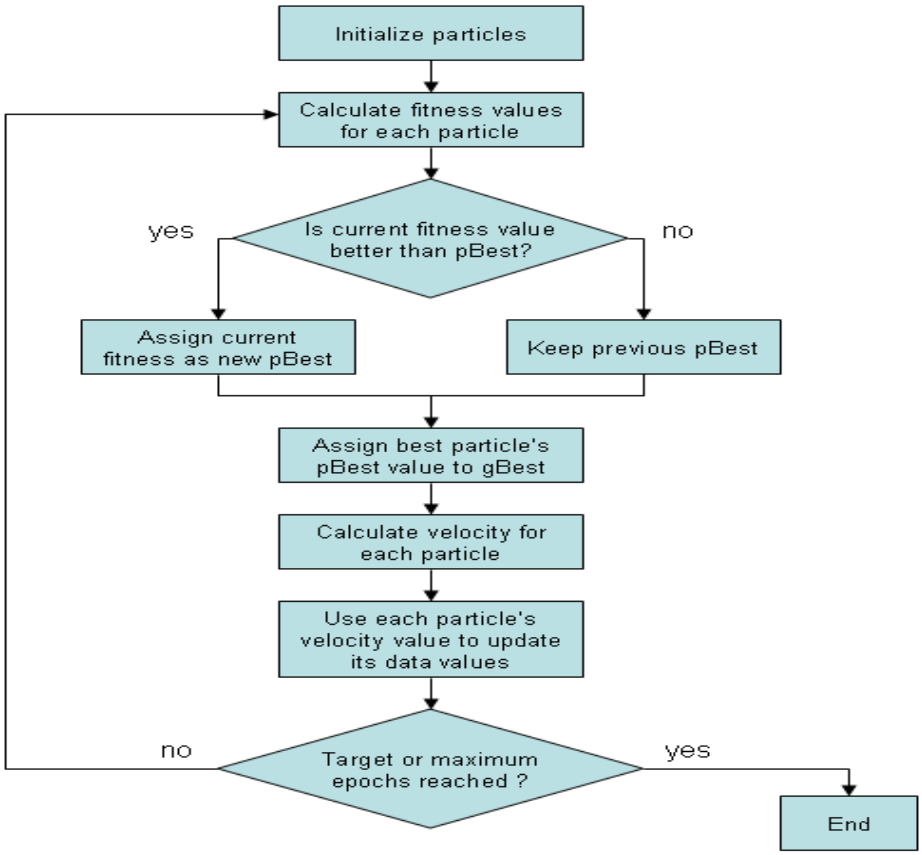

Figure 2.2: *flow chart of*PSO

DE(Differential Equation):

DE is one of the most recent EA in order to solve real-parameteroptimization problems. Thebasic concept of DE algorithm is having basic variant that works by having a population of candidate solution (called agents). These agents are moved around in the search-space by usingsimple mathematical function to combine the positions of existing agents from the population. Ifthe new position of an agent is an improvement it is accepted and forms part of the population, otherwise the new position is simply discarded. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered. Formally, let $f: \mathbb{R}^n \to \mathbb{R}$be the cost function which must be minimized or fitness function which must be maximized [1].

Algorithm of DE:

DE

1. Generate an Initial Population P^G
2. ***Evaluate*** P^G
3. For each individual I in P^G
4. ***Reproduce*** an offspring J from I
5. $P^{G+1} = P^{G+1} \cup$ ***Select*** (I, J)
6. Set G = G+1
7. Repeat Step 3 to 6 until termination criteria is met

2.2.2 Multiple Object Optimizations:

In our everyday life, we make decisions consciously or unconsciously. These decisions can be very simple, such as selecting the color of a dress or deciding the menu for lunch, or may be as difficult as those involved in designing a missile or in selecting a career. The former decisions are easy to make, while the latter might take several years due to the level of complexity involved. The main goal of most kinds of decision-making is to optimize one or more criteria in order to achieve the desired result. In other words, problems related to optimization abound in real life. A multi-objective optimization with conflicting objectives, there is no single optimal solution. The interaction among different objectives gives rise to a set of compromised solutions, largely known as the trade-off, non-dominated, non-inferior or Pareto-optimal solution.

2.3.1Evolutionary Algorithm:

Evolutionary algorithm applies the principles of evolution found in nature to the problem of finding an optimal solution to a solver problem. It is also called genetic algorithm. In a "genetic algorithm," the problem is encoded in a series of bit strings that are manipulated by the algorithm; in an "evolutionary algorithm," the decision variables and problem functions are used directly. Most commercial Solver products are based on evolutionary algorithms.In artificial intelligence, an evolutionary algorithm (EA) is a subset of evolutionary computation, a generic population-based Meta heuristic optimizationalgorithm. An EA uses mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, and selection. Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the quality of the solutions (see also loss function). Evolution of the population then takes place after the repeated application of the above operators. *Artificial evolution* (AE) describes a process involving individual *evolutionary algorithm*s; EAs are individual components that participate in an AE.In artificial intelligence, an evolutionary algorithm (EA) is a subset of evolutionary computation, a generic population-based Metaheuristic optimization algorithm.An evolutionary algorithm for optimization is different from "classical" optimization methods in several ways:

* Population

*Selection

*CrossOver

*Mutation

2.3.2Population &Selection:

Population where most classical optimization methods maintain a single best solution found so far, an evolutionary algorithm maintains a population of candidate solutions. Only one (or a few, with equivalent objectives) of these is "best," but the other members of the population are "sample points" in other regions of the search space, where a better solution may later be found.Another term of GA is selection which inspired by the role of natural selection in evolution – an evolutionary algorithm performs a selection process in which the "most-fit"

members of the population survive, and the "least fit" members are eliminated. In a constrained optimization problem, the notion of "fitness" depends partly on whether a solution is feasible (i.e. whether it satisfies all of the constraints), and partly on its objective function value. The selection process is the step that guides the evolutionary algorithm towards ever-better solutions.

2.3.3 Crossover &Mutation:

Crossover inspired by the role of mutation of an organism's DNA in natural evolution -- an evolutionary algorithm periodically makes random changes or mutations in one or more members of the current population, yielding a new candidate solution (which may be better or worse than existing population members).There are many possible ways to perform a "mutation," and the Evolutionary Solver actually employs three different mutation strategies. The result of a mutation may be an infeasible solution, and the Evolutionary Solver attempts to "repair" such a solution to make it feasible; this is sometimes, but not always, successful.Mutation inspired by the role of sexual reproduction in the evolution of living things -- an evolutionary algorithm attempts to combine elements of existing solutions in order to create a new solution, with some of the features of each "parent." The elements (e.g. decision variable values) of existing solutions are combined in a "crossover" operation, inspired by the crossover of DNA strands that occurs in reproduction of biological organisms.As with mutation, there are many possible ways to perform a crossover operation -- some much better than others -- and the Evolutionary Solver actually employs multiple variations of two different crossover strategies.

Simplex Crossover Operator (SPX):

SPX is a multi-parent operator, allowing a user-defined number of parents and offspring. The parents form a convex hull, called a simplex. Offspring are generated uniformly at random from within the simplex[18]. The expansion rate parameter can be used to expand the size of the simplex beyond the bounds of the parents. For example, the figure below shows three parent points and the offspring distribution, clearly filling an expanded triangular simplex[17].

UNDX:

Unimodal Normal Distribution crossover is generates of spring by using the normal distribution which is defined by μ parent. Offspring are generated around the mean; the probability of creating an offspring away from the mean vector reduces,preserves the correlation among parameters well – efficiently solves problem with strong epitasis among parameters, there can be some areas where it cannot generate offspring from a given initial population, the complexity of creating one offspring is $O(\mu^2)$ has difficulties in finding the optimal point(s) near the boundaries.

PCX:

PCX is known as Parent-Centric Crossover where offspring are centered on each parent; it assigns more probability for an offspring to remaincloser to the parents than away from parents, the complexity for creating one offspring is $O(\mu)$.

2.4.1: Swarm Intelligence:

Swarm intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial[1]. It is a relatively new discipline that deals with the study of self-organizing processes both in nature and in artificial systems. Researchers in etiology and animal behavior have proposed many models to explain interesting aspects of social insect behavior such as self-organization and shape-formation. The inspiration often comes from nature, especially biological systems. Recently, algorithms inspired by these models have been proposed to solve difficult computational problems [2].This concept was introduced by Gerardo Beni and Jing Wang in 1989, in the context of cellular robotic systems. This expression use widely in artificial inelegancy. SI systems consist typically of a population of simple agentsor boids interacting locally with one another and with their environment. The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergenceof "intelligent" global behavior, unknown to the individual agents. Examples in natural systems of SI include ant colonies, bird flocking, animal herding, bacterialgrowth, fish schooling and microbial intelligence.Some human artifacts also fall into the domain of swarm intelligence, notably some multi-robot systems, and also certain computer programs that are written to tackle optimizationand data analysis problems.

CHAPTER 3

Basic Ant Colony Optimization:

Ant colony optimization was introduced by M. Dorigo in 1990 for hard combinatorial Optimization problems [20, 22, and 23]. This Algorithm is used to solve single optimization problem with reasonable amount of computational time. It is introduced based on the behavior of real ants. When ants are searching for food they initially explore the area near their nest in a random manner. If some of them find them find the food source with good amount of food they came back to the nest with some of the food and they deposit chemical pheromone trail while they came back. The quantity of pheromone deposits lay on the path they find the food source is the way of the other ant to find that source easily. This is the indirect way of communication between the ants via pheromone trails which help them to find the shortest path between the nest and food source. If the two group of ants find the same sources with two paths, the next ant from the nest will choose the shortest path from that two path by the justify the amount of pheromone trails because the amount of pheromone evaporate with time so the shortest path will carry the more pheromones than the longest path. This characteristic of the real ant colonies are exploited in artificial ant colonies to solve the optimization problems. In Ant colony optimization algorithm is parameterized probabilistic model which is called Pheromone model.

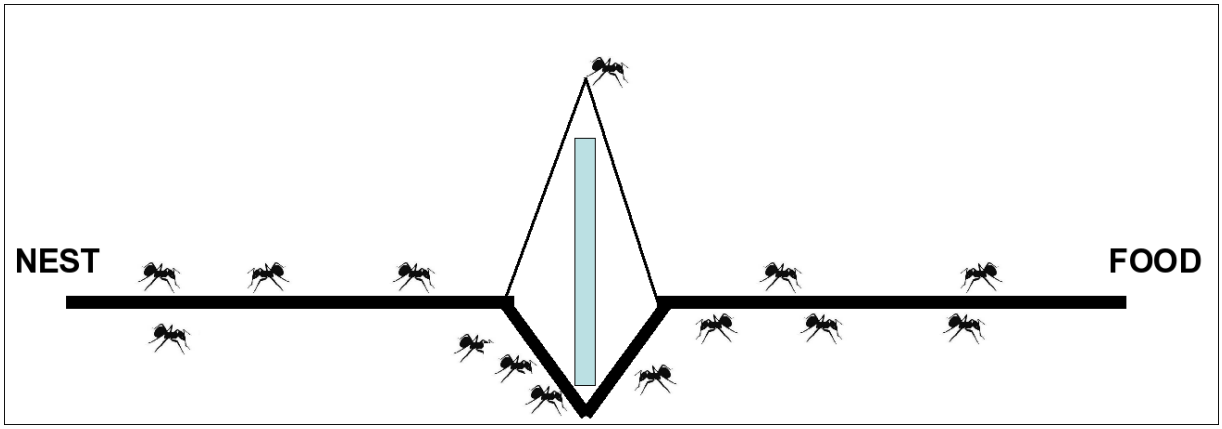

Figure: 3.1. ACO model

ACO algorithms are stochastic search procedures. The pheromone model of ACO is the probabilistic model of search space.

3.1The Basic Model of ACO Algorithms:

Ant colony optimization is a technique, which solve problems byfinding good path through graphs. Each ant from the ant colony tries to find the good path between its nest and the food sources.

```
1  procedure ACO_Meta_heuristic()
2    while (termination_criterion_not_satisfied)
3      schedule_activities
4        ants_generation_and_activity();
5        pheromone_evaporation();
6        daemon_actions();   {optional}
7      end schedule_activities
8    end while
9  end procedure

10 procedure ants_generation_and_activity()
11   while (available_resources)
12     schedule_the_creation_of_a_new_ant();
13     new_active_ant();
14   end while
15 end procedure

16 procedure new_active_ant() {ant lifecycle}
17   initialize_ant();
18   M = update_ant_memory();
19   while (current_state ≠ target_state)
20     A = read_local_ant-routing_table();
21     P = compute_transition_probabilities(A, M, problem_constraints);
22     next_state = apply_ant_decision_policy(P, problem_constraints);
23     move_to_next_state(next_state);
24     if (online_step-by-step_pheromone_update)
25       deposit_pheromone_on_the_visited_arc();
26       update_ant-routing_table();
27     end if
28     M = update_internal_state();
29   end while
30   if (online_delayed_pheromone_update)
31     evaluate_solution();
32     deposit_pheromone_on_all_visited_arcs();
33     update_ant-routing_table();
34   end if
35   die();
36 end procedure
```

3.1.1 Generate Solution:

This procedure includes the routines needed for ants to construct solutions incrementally. After the selection of the starting city, one node at a time is added to an ant's path. An ant's decision of where to go next is biased by the pheromone trails i,jand the heuristic information i,j. In general, the higher the two values, the higher the probability of choosing the associated edge. Typically, two parameters $\alpha > 0$ and $\beta \geq 0$ are used to weigh the relative influence of the pheromone and the heuristic values, respectively. The rule that defines the ant's choice is specific to each ACO variant.

3.1.2 Daemon Action:

This procedure comprises all problem-specific operations that may be considered for boosting the performance of ACO algorithms. The main example of such operations is the introduction of a local search phase. In addition, daemon actions implement centralized tasks that cannot be performed by an individual ant. This type of procedure is of optional use, but typically several daemon actions are very useful to significantly improve the performance of ACO algorithms (Dorigo and Stutzle, 2004).

3.1.3 Pheromone Update:

Artificial ants modify some environmental elements as the real ants. While real ants deposit a chemical substances called pheromone and in artificial ant colony optimization the artificial ants update their pheromone with some numerical equations. Pheromone update is very important part of Ant colony optimization because If the artificial ant competition the trial then the pheromone level will update by –

$$\tau_{xy} \leftarrow (1-\rho)\tau_{xy} + \sum_{k} \Delta\tau_{xy}^{k}$$

Here τ_{xy} is the amount of pheromone deposited in respect of x, y and ρ is the pheromone Evaporation coefficient and of course $\Delta\tau_{xy}^{k}$ is the amount of pheromone deposited by Kth ant.We use this equation to update the pheromone. When this Ant colony optimization algorithm finds a good value then the pheromone update is needed to change the value with good value.

SBX-

Simulated binary crossover mainly works to simulate the offspring distribution of single-point, mainly binary encoded crossover on real-valued decision variables. The parameter is generally for the simulated binary crossover is probabilityand distribution of the index. Here probability defines the probability of applying the simulatedBinary crossover and distribution Index means the index of the simulated binary crossoveroperator.Simulated Binary Crossover (SBX) was proposed in 1995 by Deb and Agrawal.SBX was design with respect to the one-point crossover properties in binary-coded GA.

Average Property of SBX:

The average property of the decoded parameter values is the same before and after the crossover operation Spread Factor Property: The probability of occurrence of the spread factor $\beta \approx 1$ is more likely than any other β value.β is defined as the ratio of the spread of offspring point to that in the parent points :

$\beta = |(c1\text{-}c2) \div (p1 - p2)|$

Contracting Crossover-

$\beta < 1$

The offspring points are enclosed by the parent points.

Expanding Crossover-

$\beta > 1$

The offspring points enclose by the parent points.

Stationary Crossover-

$\beta = 1$

The offspring points are the same as parent points.

PM:

Polynomial operator attempts to simulate the distribution of binary encoded bit-flip mutation on the real valued decision variables the polynomial basically favors the offspring nearer to the parents. The distribution index of polynomial mutation controls the shape of the offspring

distribution and the larger values for the distribution index mainly generates offspring closer to the parents. The operator of the PM is also take two operator like SBX and construct a polynomial mutation operator with the specified probability and the distribution index.

Flow Chart:

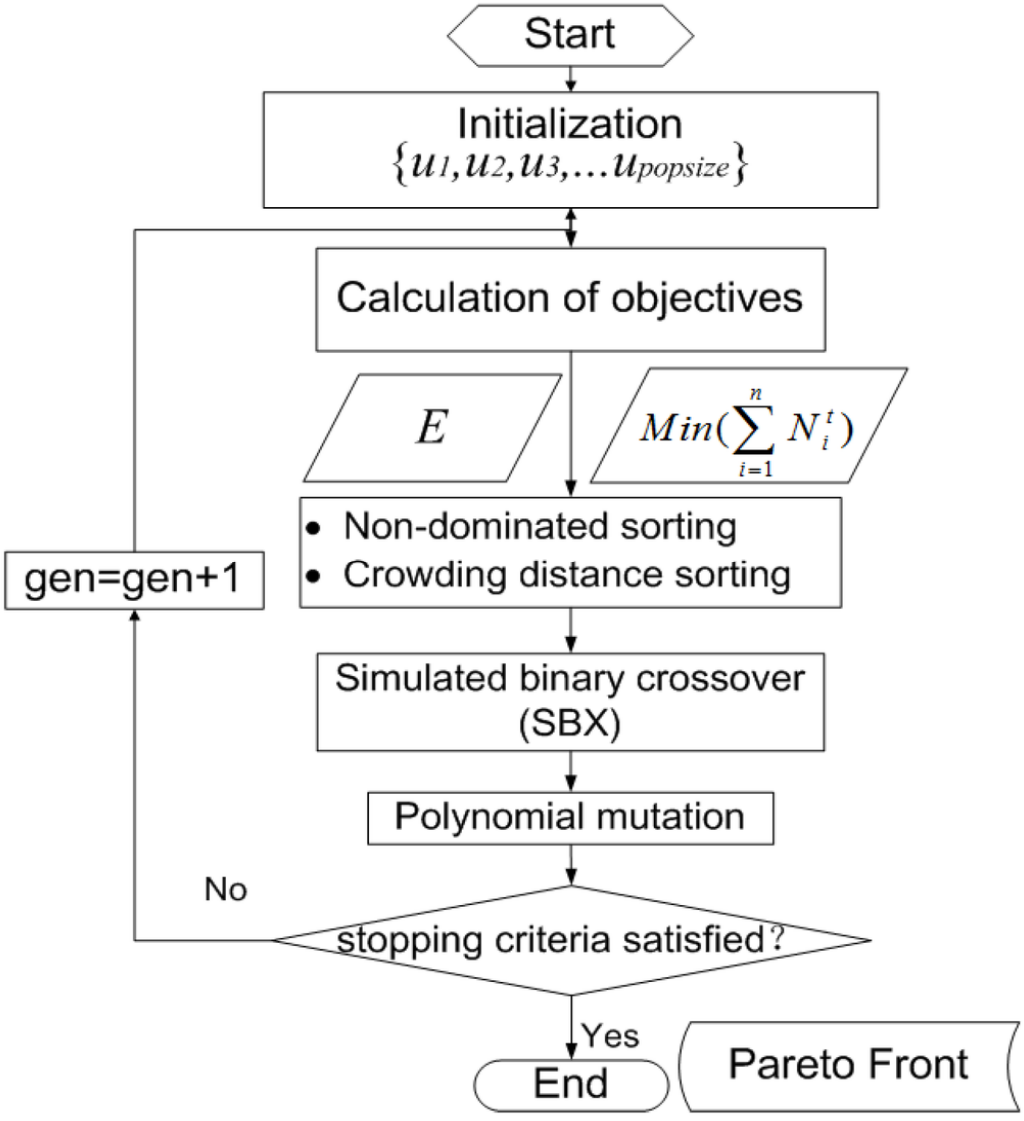

***Figure 3.2*:** Flow chart of SBX and Polynomial

3.2 SBX ACO

ACO is a probabilistic technique and search for an optimal path and it’s based on the behavior of the real ants seeking a path between their colony and source food. Basically to enhance the efficiency and for a new algorithm SBX-ACO we use simulated binary crossover, polynomial mutation and ant colony optimization algorithm and try to justify the efficiency level by benchmark function. We also use polynomial mutation for the better performance of our algorithm. It’s quite similar to SBX that's why we merge these two operators and include it on our operation for better performance. As our main target to propose a new algorithm and we choose ant colony optimization and some other operator like simulated binary crossover and polynomial mutation because when to research we found that their combination could give good solution. We choose ACO because the main goal of ant colony optimization is to improve the performance of algorithm and the second one is to investigate the better explain its behavior.

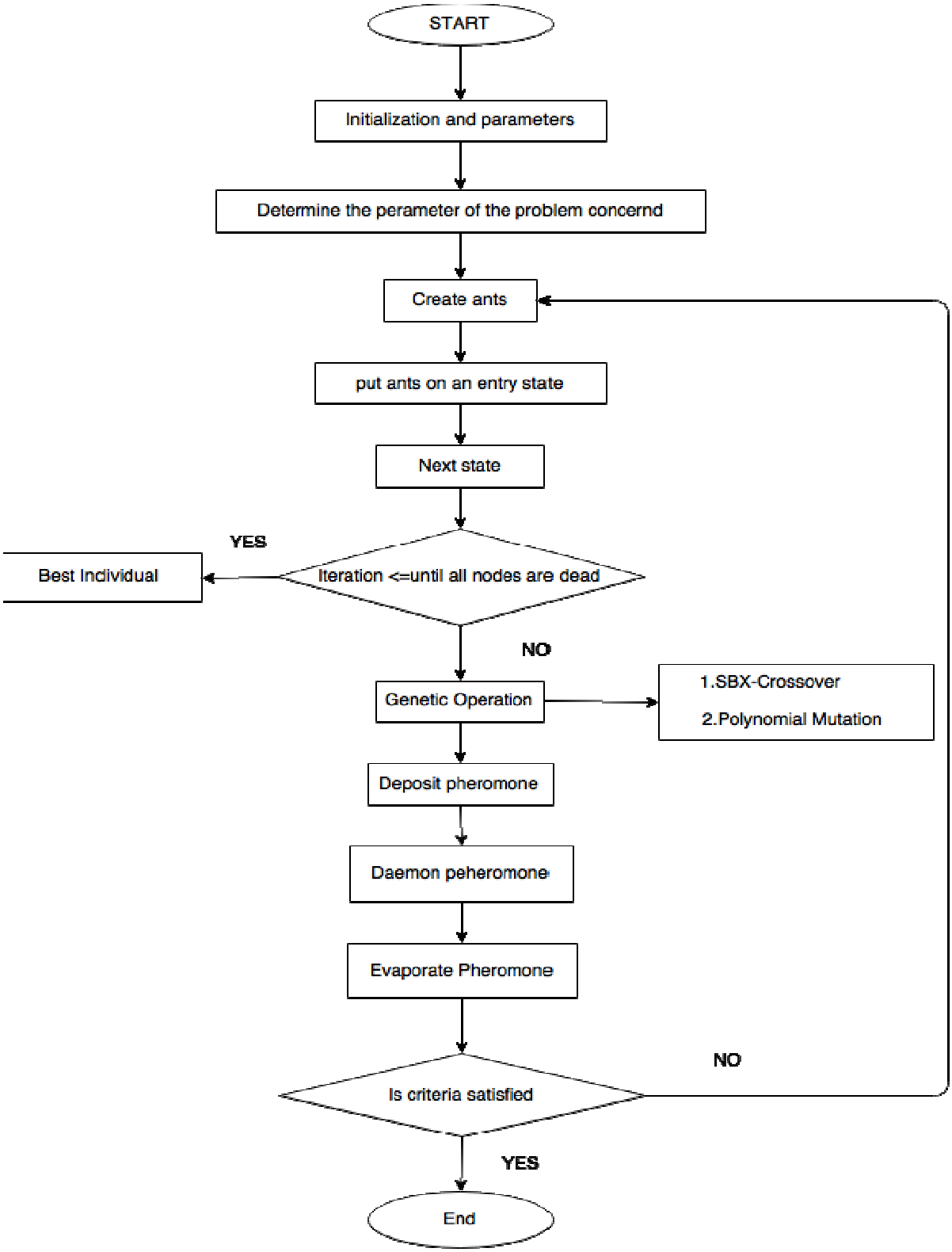

***Figure**:* 3.3: Flow chart of SBXACO

Pseudo codes:

1. procedure SBX_ACO ()

2. initialize population and parameters

3. determine the parameter of the problem concern

4. create_artificial_ants

5. put the ant on entry level

6.put the other ants on next state

6.while(Iteration is not completed)

8.while(best_indidual_not_found)

7.genetic operation-

-SBX-crossover

-Mutation

8.Deposit Pheromone

9.Daemon Actions

10.Evaporate pheromone

11.If best individual found

12.end while

13. if best individual not found then again create ants

14.end procedure

In SBX-ACO we basically initialize the population and parameters as we require for any problem and then create artificial ants and put them on the start level of the algorithms the process will continue till the iteration is not completed our target is to find the best solution from it so in every time we take the best individuals each time when we get better than it we exchange it with the previous result. If we did not find any solution then we again start or create ant from the begging of the algorithm. When we running the algorithmwe run the genetic algorithm through it we choose SBX-crossover and Mutation before pheromone update. After the pheromone update we Deposit Pheromone,Daemon Actions Evaporate pheromone as we did in ACO.Then get the best solution from it.This procedure will be running until we get the solution

3.3Advantage and disadvantage of ACO:

Advantage-

1. This algorithm is already proven that this algorithm has a positive feedback accounts for rapid discovery of good solutions

2. This algorithm ensures the interaction of a population agents.

3. ACO is ensures Distributed computation which avoids premature convergence

4. This algorithm is inherent parallelism

5. This is very much efficient for travelling salesman problem

6. This algorithm can be used in dynamic applications

Disadvantage-

1. It is not independent because it's a sequence of random decisions
2. Probability distribution change due to iteration.
3. Research is experimental rather than theoretical.
4. Convergence is guaranteed but time to convergence is uncertain

3.4 Recent Research and Advanced Topic on ACO:

There are many available successful implantation of ant colony optimization to solve many Meta heuristic problems. ACO is use to solve Travelling salesman problem. ACO is also applied to solve many other problems such as vehicle routing problems, scheduling problems. Currently state-of-the-art for solving the sequential ordering problem, project scheduling problem and open shop scheduling problem. ACO is also tried to apply on multi-object and the new algorithm for multi-object is called MACO.Ant colony optimization is use in networking because now a days it is use to solve in routing problem in telecommunication networks. An improved version of ant colony optimization is use for solving constraint satisfaction problems. For solving any problems it can be customized as the requirements and apply ACO can be give a better result. ACO is also applied in scheduling problems like it is already used in shop scheduling, project scheduling and able to solve many other scheduling problems with proper solutions. One of the other applications ACO is to draw Network models.

CHAPTER 4

Bench mark Function:

Test function is known as benchmark functions which use vastly in Computer Science. In applied mathematics, benchmark functions are known as artificial landscapes. These functions are useful to evaluate characteristics of optimization algorithms according to:

- Velocity of convergence.
- Precision.
- Robustness.
- General performance

In computer science, benchmark functions are used to check efficiency of optimization algorithm. The performance evaluates how optimal such algorithm is. It is very useful for analyzing algorithms upon constraint and unconstraint function. Recently, a paper published by Noman and Iba evaluate DE algorithm by testing upon them in benchmark Function. They also introduced a hybrid DEahcSPX and use them solving benchmark functions in order to compare performances of duo. Basically benchmark functions are mathematical functions but use as a testing composite in EA. The main reason of using this function is; these mathematical functions need to be solved by constraint, unconstraint and dynamically optimization where parameter choosing is very important. Here we choose five benchmark functions with wide range to test upon. The properties of chosen function are given below.

Function 1:Sphere

Equation:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$$

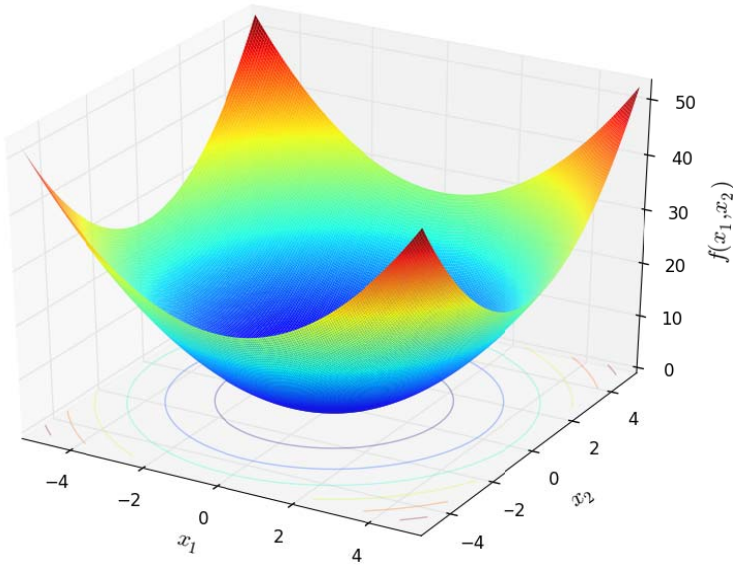

Figure4.1: Sphere function

Definition:

- Search domain: $-5.12 \leq x_i \leq 5.12,\ i = 1, 2. . . .n.$
- Number of local minima: no local minimum except the global one.
- The global minima: $x^* = (0, ..., 0), f(x^*) = 0$
- Function graph: for $n = 2$

Function 2: Rosenbrock

Equation-

$$f(\boldsymbol{x}) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$$

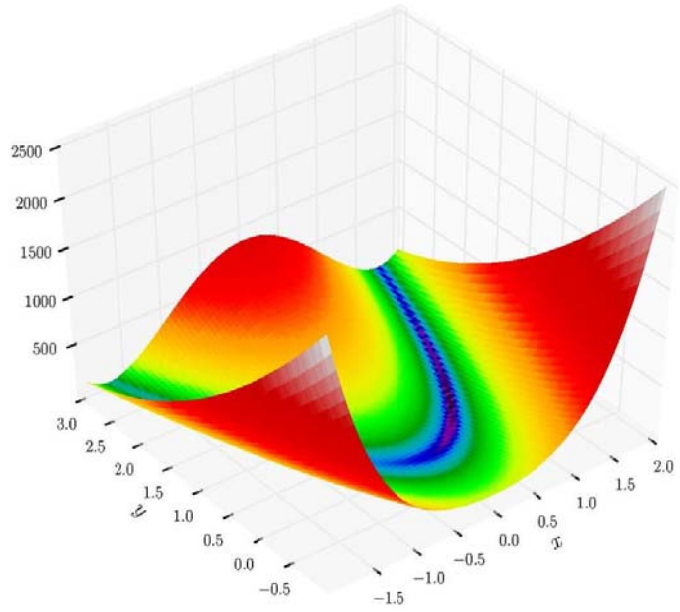

Figure 4.2: Rosenbrock function

Definition:

- Number of variables: *n* variables.
- Search domain: $-5 \leq x_i \leq 10,\ i = 1, 2. \ldots n.$
- Number of local minima: several local minima.
- The global minima: $x^* = (1 \ldots 1),\ f(x^*) = 0.$
- Function graph: for $n = 2$.

Function 3: Scaffer:

Equation:

$$f(x,y) = 0.5 + \frac{\sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$$

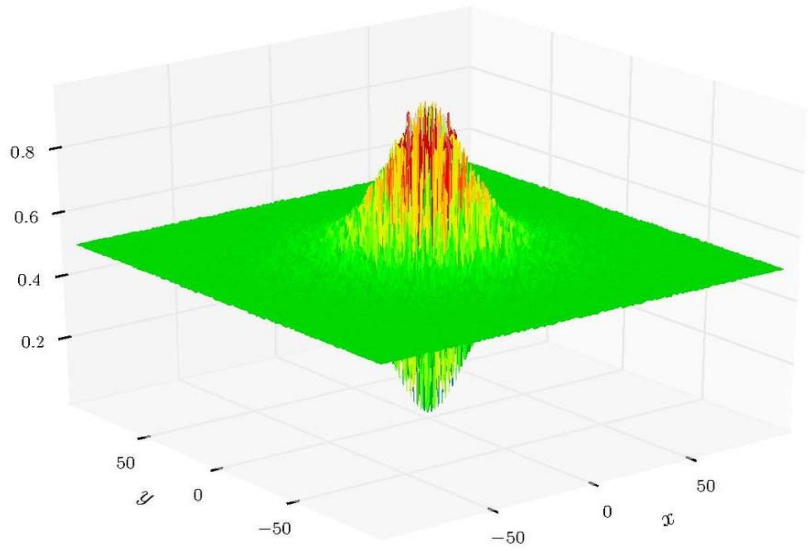

Figure 4.3: Scaffer function

Definition:

- The function is usually evaluated on the square $x_i \in$ [-100, 100], for all i = 1, 2.
- $f(x^*) = 0$ $x^* = (0,0)$
- Number of local minima: several local minima.
- Function graph: for $n = 2$.

Function 4: Griewank

Equation:

$$1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$$

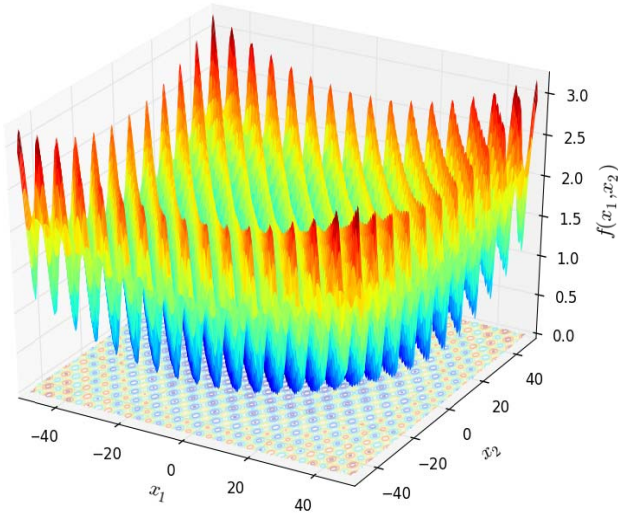

Figure 4.4: Griewank function

Definition:

- Number of variables: *n* variables.
- Search domain: $-600 \leq x_i \leq 600,\ i = 1, 2, \ldots, n.$
- Number of local minima: several local minimum.
- The global minima: $x^* = (0, \ldots, 0), f(x^*) = 0.$
- Function graph: for $n = 2$.

Function 5:Ackley

$$f(x,y)= -20 \exp\left(-0.2\sqrt{0.5((x^2 + y^2))}\right) - \exp\left(0.5(\cos(2\pi x) + \cos(2\pi y))\right)+\exp+20$$

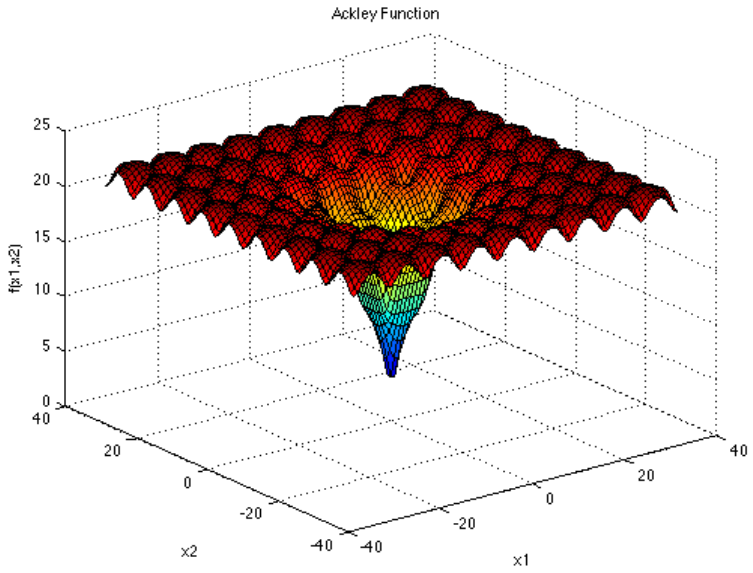

Figure 4.5:Ackley function

Definition:

- Number of variables: *n* variables.
- Search domain: $-15 \leq x_i \leq 30,\ i = 1, 2, \ldots, n.$
- Number of local minima: several local minima.
- The global minimum: $\mathbf{x}^* = (0, \ldots, 0), f(\mathbf{x}^*) = 0.$
- Function graph: for $n = 2$.

CHAPTER 5

Experiments:

Our goal was to make a hybrid of single object optimization algorithm ACO. We use SBX crossover and proposing a new hybrid named SBXACO. We have carried out different experiment to assess the performance of SBXACO using the test function described in *chapter 4*. Given five test functions are functions commonly found in the literature for CEC 2005 Special Session on real-parameter optimization [9]. The focus of the study was to compare the performance of the proposed SBXACO algorithm with the original ACO algorithm in different experiment.

5.1 Performance evaluation Criteria:

For evaluating the performance of the algorithms, several of the performance criteria of [9] were used with the difference that 50 instead of 25 trails were conducted, respectively. We compared the performance of SBXACO with ACO for the test suite using the *function error value*. The maximum number of fitness evaluations that we allowed for each algorithm to minimize this error was 20000**N*, where is *N*the dimension of the problem. The fitness evaluation criteria were as follows.

Evaluation:

The number of function evaluations (FEs) required reaching an error value less than (provided that recorded in different runs and the average and standard deviation of the number of evaluations were calculated. For the functions F_1 to F_2, the accuracy level was fixed at 10^{-6}.For this criterion, the notation was used AVG SD by which the algorithm could reach the accuracy level.

5.1.1 Convergence graphs:

Convergence graphs of the algorithms. These graphs show the average **FE** performance of the total runs, in respective experiments.

5.1.2Experimental Setup:

In our experimentation, we used the same set of initial random populations to evaluate different algorithms in a similar way done in [1].Though classic ACO uses only three control parameters namely *Update Pheromone, Demon Action* and *Evaporation.* We use SBX operator for crossover and PM for mutation. For the SBX operation, we chose the number of parents participating in the crossover operation to be $x_p = 0.9$ as suggested in [11] and changes to this setting are also examined.The experiments were performed on a computer with 4400 MHz AMD Athlon TM 64 dual core processors and 2 GBof RAM in Java 2 Runtime Environment.

5.1.3Comparison with Basic ACO:

Most of the cases (testing upon function) we get convincing result of our SBXACO rather than ACO. We could not find better result than ACO only in Ackley function. The table 5.1.1 and graph*fig:(5.1.1-5.1.5)* is stated on to show the comparison in below. In graphical comparison it has shown that ACO performed less efficiently in terms of optimization rather than our proposed SBXACO.

5.1.4 Comparison with DE and DEachSPX:

In our experiment we take value of DE and DEachSPX from an existing literature publish by IEEE[1].While comparing our result with their value we found that our hybrid SBX. Some function were not tested by DE and DEachSPX in the mentioned literature[1] so we ignore those values ACO shows better performance than DE but DEachSPX works well most of the cases then us.

Function	Object	Range
Function01	Minimize	$[-5.12,5.12]^{30}$
Function02	Minimize	$[-2.048,2.048]^{30}$
Function03	Minimize	[-100,100]
Function04	Minimize	$[[-600,600]^{10}$
Function05	Minimize	[-32,32]

Table: 5.1.1

FUNCTION	ACO	SBX-ACO	DE	DEachSPX
1.Sphere	15073.6 ±9461.3	6090.0 ± 9499.9(50)	148650.8 ±6977.7 (50)	87027.1± 3967.3 (32)
2.Rosenbork	588987.5 ±444727.9 (50)	535122.9± 361327.2	-	299913.0 ±519.5 (48)
3.Scaffer	13559.9 ±8397.8(50)	5102.7±4325.1 (50)	-	-
4.Grienwank	16014.9 ±6758.4(50)	57202.3± 15183.1(50)	190292.5±63478.8 (40)	121579.2±79563.4 (43)
5.Ackleys	206306.1 ±12177.6(50)	202856.2 ±10235.5(50)	215456.1±9721.4 (2)	129211.6±518.6 (45)

Table: 5.1.2

In following graphs FE(Functional Evolution) lies on X-axis and Itterartion on Y-axis

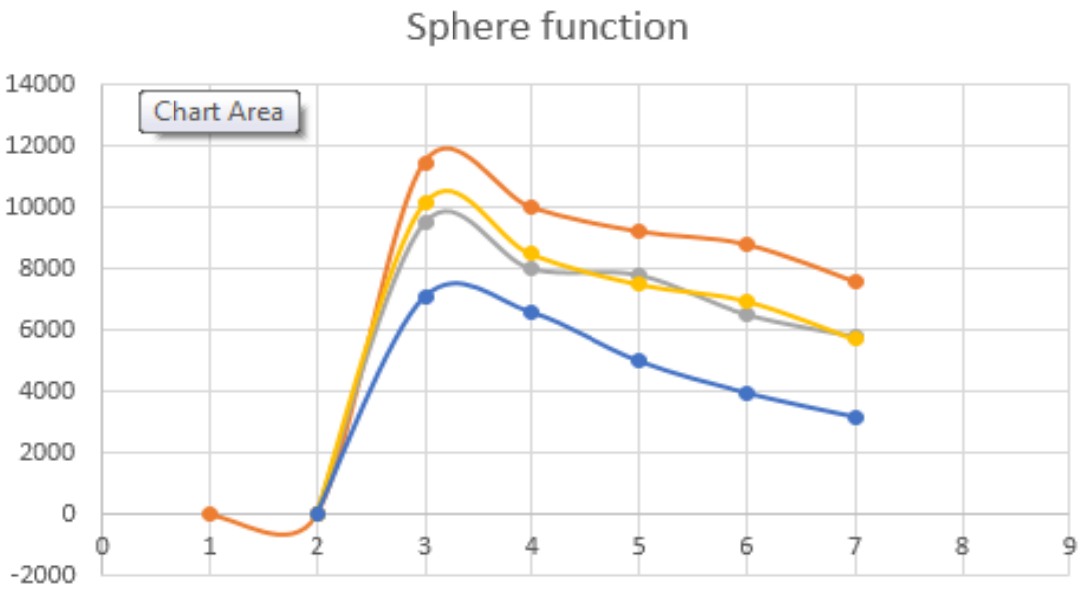

Figure: 5.1.1

Here in (fig:5.1.1)Sphere function SBXACO(Grey) has good result than ACO(Red) and DE(yellow) but DEachSPX (Blue) which perform better then all.

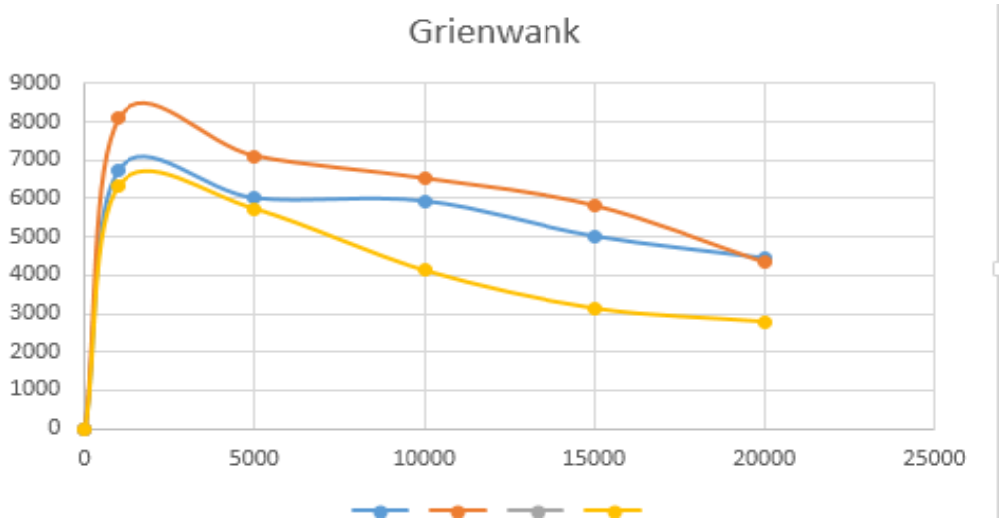

Figure: 5.1.2

Here in (fig:5.1.2)Grienwank function SBXACO(Yellow) has good result than ACO(Blue) and DE(yellow) but DEachSPX (Red) which perform better then all.

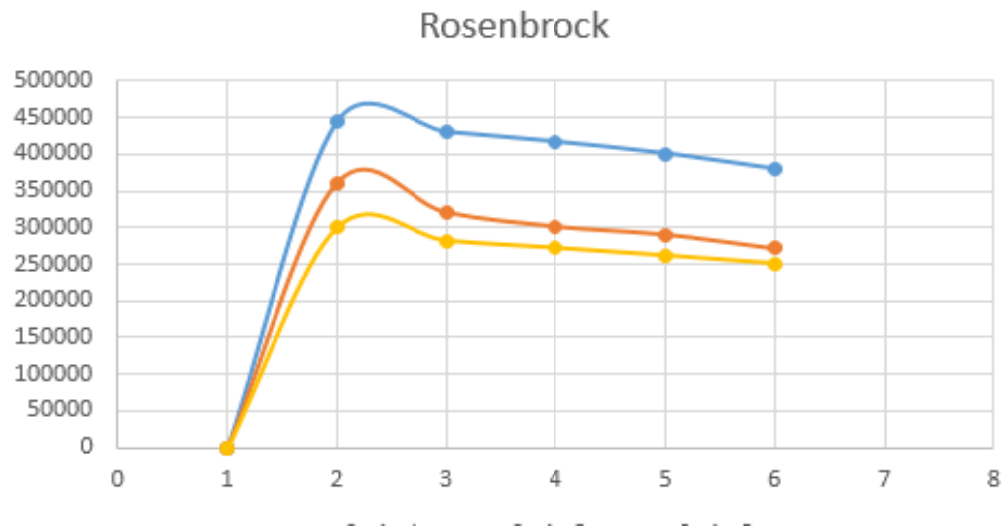

Figure: 5.1.3

Here in (fig:5.1.3)X-axis is equal 5000 .Rosenbrock function SBXACO(Yellow) has good result than ACO(Red) but DEachSPX (Blue) which perform better then all.

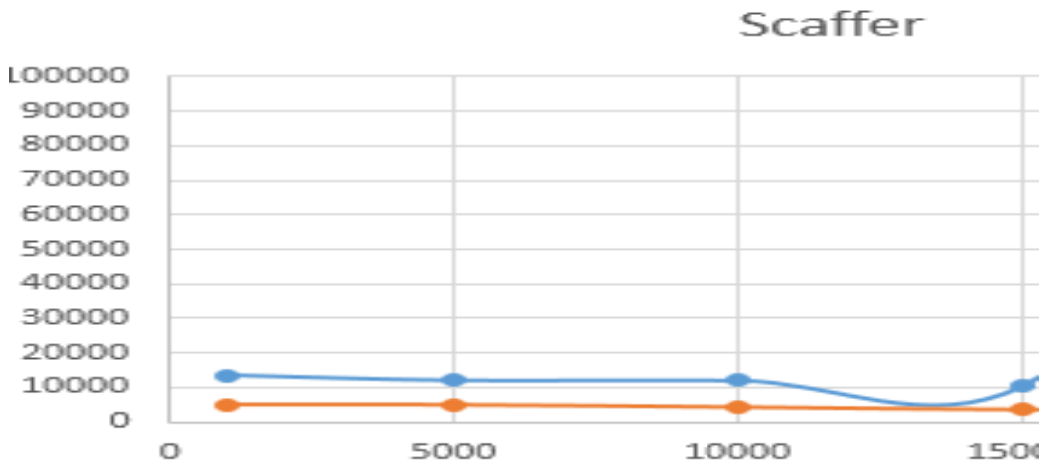

Figure: 5.1.4

Here in (fig:5.1.4)Scaffer function SBXACO(Red) has good result than ACO(Blue) .We could not find DE and DEachSPX value in mentioned[1] literature.

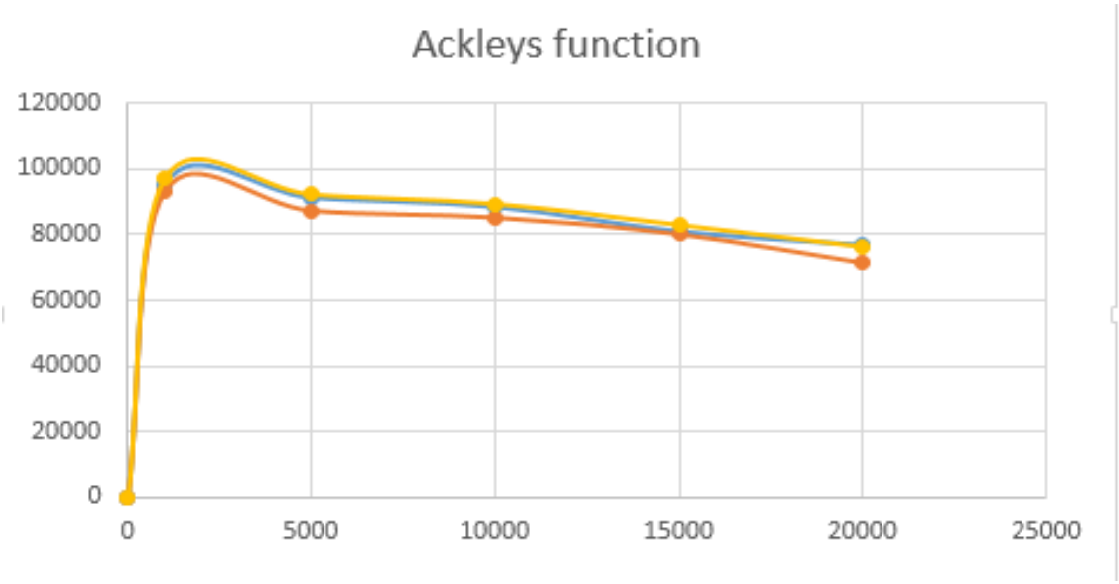

Figure: 5.1.5

Here in (fig:5.1.1)Ackleys function SBXACO(Grey) has good result than ACO(Yellow)but DEachSPX (Blue) which perform better then all.

CHAPTER 6

Application:

This chapter discusses the various application areas of ACO method.This algorithm is a member of the ant colony algorithms family, in swarm intelligence methods, and it constitutes some Metaheuristic optimizations. Initially proposed by Marco Dorigo in 1992 in his PhD thesis, the first algorithm was aiming to search for an optimal path in a graph [8], based on the behavior of ants seeking a path between their colony and a source of food. The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems have emerged, drawing on various aspects of the behavior of ants. There are huge field in both soft

computing to hardware where ACO can use .More over our hybrid SBXACO also can be implemented for real life problem.

Design and Modeling	Conceptual design, electromagnetics case, induction heating cooker design, VLSI design, power systems, RF circuit synthesis, worst case electronic design, motor design, filter design, antenna design, CMOS wideband amplifier design, logic circuits design, transmission lines, mechanical design, library search, inversion of underwater acoustic models, modeling MIDI music, customer satisfaction models, thermal process system identification, friction models, model selection, ultrawideband channel modeling, identifying ARMAX models, power plants and systems, chaotic time series modeling, model order reduction.
Biomedical	Human tremor analysis for the diagnosis of Parkinson's disease, inference of gene regulatory networks, human movement biomechanics optimization, RNA secondary structure determination,

	phylogenetic tree reconstruction, cancer classification, and survival prediction, DNA motif detection, biomarker selection, protein structure prediction and docking, drug design, radiotherapy planning, analysis of brain magneto encephalography data, electroencephalogram analysis, biometrics and so on
Networking	Radar networks, bluetooth networks, auto tuning for universal mobile telecommunication system networks, optimal equipment placement in mobile communication, TCP network control, routing, wavelength division-multiplexed network, peer-to-peer networks, bandwidth and channel allocation, WDM telecommunication networks, wireless networks, grouped and delayed broadcasting, bandwidth reservation, transmission network planning, voltage regulation, network reconfiguration and expansion, economic dispatch problem,

	distributed generation, micro grids, congestion management, cellular neural networks, design of radial basis function networks, feed forward neural network training, product unit networks, neural gas networks, design of recurrent neural networks, wavelet neural networks, neuron controllers, wireless sensor network design, estimation of target position in wireless sensor networks, wireless video sensor networks optimization
Robotics	Control of robotic manipulators and arms, motion planning and control, odour source localization, soccer playing, robot running, robot vision, collective robotic search, transport robots, unsupervised robotic learning, path planning, obstacle avoidance, swarm robotics, unmanned vehicle navigation, environment mapping, voice control of robots, and so forth.

Image and Graphics	Planning landmarks in orthodontic x-ray images, image classification, inversion of ocean color reflectance measurements, image fusion, photo time-stamp recognition, traffic stop-sign detection, defect detection, image registration, microwave imaging, pixel classification, detection of objects, pedestrian detection and tracking, texture synthesis, scene matching, contrast enhancement, 3D recovery with structured beam matrix, character recognition, image noise cancellation.
Fuzzy systems, Clustering, data mining	Design of neurofuzzy networks, fuzzy rule extraction, fuzzy control, membership functions optimization, fuzzy modeling, fuzzy classification, design of hierarchical fuzzy systems, fuzzy queue management, clustering, clustering in large spatial databases, document and information clustering, dynamic clustering, cascading classifiers, classification of hierarchical biological data, dimensionality reduction, genetic-

	programming-based classification, fuzzy clustering, classification threshold optimization, electrical wader sort classification, data mining, feature selection
Optimization	Electrical motors optimization, optimization of internal combustion engines, optimization of nuclear electric propulsion systems, floor planning, travelling-sales man problems, n-queens problem, packing and knapsack, minimum spanning trees, satisfiability, knights cover problem, layout optimization, path optimization, urban planning, FPGA placement and routing.
Prediction and forecasting	Water quality prediction and classification, prediction of chaotic systems, streamflow forecast, ecological models, meteorological predictions, prediction of the floe stress in steel, time series prediction, electric load forecasting, battery pack state of charge estimation, predictions of elephant migrations,

	prediction of surface roughness in end milling, urban traffic flow forecasting and so on.

CHAPTER 7

Conclusion:

This thesis discussed the basic Ant colony Optimization algorithm, geometrical and mathematical explanation of ACO, demon action, pheromone update, evaporation in the search space. Function and parameter are also explained in chapter 3.SBX and Polynomial mutation also described in this chapter. For better explanation flowcharts and pseducode are also mentioned. Our proposed hybrid SBXACO is also described here with necessary functions.

In chapter 4, our chosen benchmark functions are defined with graph. Here the explanations have been given that why benchmark function isused to evaluate performances of algorithm.

In chapter 5, Experiments are explained with comparable table and their graph in order to display the comparisons clearly. Here we state another two algorithm DE and DEachSPX from an existing literature by Noman and Iba. We use theirresults to compare with our proposed algorithm.

Basically we try to use GA's crossover mutation operator and make a hybrid ACO. Due to justify the efficiency we run it upon some benchmark functions. Later we compare the value with original ACO which also been tested upon benchmark functions by us.

Future plan:

In this paper, we emphasize on single objective ACO. Though we get convincing result we will submit our paper in journal. Here we work with five benchmark functions,we are working on test ACO and SBXACO with upon twenty benchmark functions In future we have plan to work with multi objective ACO also. Moreover PSO is another single objective optimization algorithm which actually can be use along with ACO to introduce a hybrid. We are eager to work with PSO. Moreover, this thesis can be used in various applications like networking, graphics, bioinformatics, robotics,design and modeling etc. We are also eager to implement our algorithm in an application in future.

References:

[1]N.Nasimul and H.Iba *Accelerating Differential Evolution Using Adaptive Local Search*,IEEE vol.12,no:01,pp 107-124,2008.

[2]Patel,K.,Kabat,R and Tripathy.2013.A hybrid ACO/PSO based algorithm for QoS multicast routing problem,inIndia,Dept. of Comp. Sc. &Engg.,Veer SurendraSai University of Technology;inAin Shams Engineering Journal,Ain Shams University:113-120.

[3]T.Satyobroto,"Mathmatical Modelling and Application of Particle Swarm Optimization",School of Engineering,Blekinge Institute of Technology;.

[4] Andries P. Engelbrecht, *Computational Intelligence: An Introduction*.: John Wiley and Sons, 2007, ch. 16, pp. 289-358

[5] Singiresu S. Rao, *Engineering Optimization Theory and Practice*, 4th edition, Ed.: John Wiley and Sons, 2009.

[6]Singiresu S.Rao,*Engineering Optimisation Theory And Practiceth*,4th edition,Ed:John Wiley and Sons,2009

[7]El-Ghazali Talbi,*Metahurestic-Form Design to Implementation*::John Wiley and Sons,2009.

[8]Dorigo, M., Birattari, M., and Stutzle, T., *Ant colony optimization,* IEEE Computational Intelligence .Magazine, 1(4):28-39, 2006.

[9]P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger,and S. Tiwari,

"Problem Definitions and Evaluation Criteria for theCEC 2005 Special Session on Real-Parameter Optimization," NanyangTechnol. Univ., Singapore, IIT Kanpur, India, KanGAL Rep. 2005005,May 2005

[10]Noman, Iba and Hitoshi *;Member, IEEE*.2008 . Accelerating Differential Evolution Using an Adaptive Local Search; in IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION,12(1):108-124.

[11] S. Tsutsui, M. Yamamura, and T. Higuchi, "Multi-parent recombination with simplex crossover in real coded genetic algorithms," in *Proc.Genetic Evol. Comput. Conf. (GECCO'99)*, Jul. 1999, pp. 657–664.

[12] Mukharjee.D and Acharyya.S.Ant Colony Technique applied in Network Routing Problem, in West Bengal University of Technology;in International journal of Computer Applications 1(15).

[13] Suganthan,P.N,Hasen.N,Liang,J.J,Deb.K,Auger.A and Tiwari.S,2005.Problem Difinatiions and Evaluation Criteria for the CEC Special Session on Real-Parameter Optimization.Nayanyang Technological University,Singapore,School of EEE.

[14]C. M. Bishop. Neural Networks for Pattern Recognition. Oxford University Press,

1995.

[15] J. Branke, M. Middendorf, and F. Schneider. Improved heuristics and a genetic algorithm for finding short supersequences. OR-Spektrum, 20:39–46, 1998.

[14]P.W .Tsai, J.S.Pan, B. Y. Liao and S.C.Chu,*Enhanced Artificial Bee Colony Optimization*,ICIC International,ISSN 1349-4198,Cheng Shiu University,2009.

[16] D. Karaboga, *An Idea Based On Honey Bee Swarm For Numerical Optimization*, *Technical Report-TR06*, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[17] K. W. Kim, M. Gen, and M. Kim, *Adaptive Genetic Algorithms for Multi-resource Constrained Project Scheduling Problem with Multiple Modes*, International Journal of Innovative Computing, Information and Control, vol.2, no.1, pp.41-49, 2006.

[18]Tsutsui, S., Yamamura, M., and Higuchi, T., "Multi-parent Recombination with Simplex Crossover in Real Coded Genetic Algorithms," Proceedings of the Genetic and Evolutionary Computation Conference, vol. 1, pp. 657-664, 1999.

[19]Higuchi, T., Tsutsui, S., and Yamamura, M., "Theoretical Analysis of Simplex Crossover for Real-Coded Genetic Algorithms," Parallel Problem Solving from Nature PPSN VI, pp. 365-374, 2000.

[20]Savic.D, “Single – Objective vs Multiobjective optimization for Integrate Decision Support”.University of Exeter,UK.Department of Engineering School and ComputerScience

[21]Bandyopadhaye,S; Saha,S;” ”.Classical and Metahurestic Approaches and Applications,pp.262,2013.

[22] M. Fatih Tasgetiren and Yun-Chia Liang, "A Binary Particle Swarm Optimization Algorithm for Lot Sizing Problem," *Journal of Economic and Social Research*, vol. 5, no. 2, pp. 1-20, 2003.

[23] D. Corne, M. Dorigo, and F. Glover, editors. New Ideas in Optimization. McGrawHill, 1999.

[24] D. Costa and A. Hertz. Ants can colour graphs. Journal of the Operational Research Society, 48:295–305, 1997.

[25] D. Costa, A. Hertz, and O. Dubuis. Embedding of a sequential algorithm within an evolutionary algorithm for coloring problems in graphs. Journal of Heuristics, 1:105–128, 1995.

[26] G.A. Croes. A method for solving traveling salesman problems. Operations Research, 6:791–812, 1958.

[27] J.-L. Deneubourg, S. Aron, S. Goss, and J.-M. Pasteels. The self-organizing exploratory pattern of the argentine ant. Journal of Insect Behavior, 3:159–168, 1990.

[28] G. Di Caro and M. Dorigo. AntNet: A mobile agents approach to adaptive routing. Technical Report 97-12, IRIDIA, Universit´e Libre de Bruxelles, 1997.

[29] G. Di Caro and M. Dorigo. Mobile agents for adaptive routing. In Proceedings of the 31st International Conference on System Sciences (HICSS-31), volume 7, pages 74–83. IEEE Computer Society Press, 1998.

[30] Muhlenbein, R. , Schom isch, M., and Born, J. , "T he Par allel Genetic Algorit hms as Function Optimizer, " in *Proceedings of the FOU1,th International Conference on Genetic Algorithms,* edited by R. K, Belew and L. B. Booker (Morgan Kaufmann, San Mateo, 1991).

[31] Gordon, V. S. and Whitley, D., "Serial and Par allel Genetic Algorithms as Function Optimizers ," in *Proceedings of the Fifth International Conference on Genetic Algorithms,* edite d by S. Forrest (Mor gan Kaufmann, San Mat eo 1993) .

[32] Deb , K. , "Genetic Algorithms in Multimodal Funct ion Optimization," TCGA Repo rt No. 89002 (Uni versity of Alabama , Tus caloosa , 1989).

[33] Gold berg , D. E., an d Richardson , J ., "Genet ic Algori thms with Sharing for Multimod al Function Optimizati on ," in *Proceedings of the Second International Confe rence on Geneti c Algor-ithms,* edited by J. J. Grefenste tte (Morgan Kaufmann, San Mateo, 1987).

[34] Srinivas , N. an d Deb , K. , "Multiobj ecti ve Function Optimization using Nondominate d Sorting in Genet ic Algorit hms," *Evolutionary Computation, 2(3) ,* 1995, 221-248.