Tribhuvan University

Institute of Science and Technology

COMPARATIVE ANALYSIS OF PARTICLE SWARM OPTIMIZATION VARYING THE INERTIA FACTOR

Dissertation

Submitted to:
Central Department of Computer Science and Information Technology
Tribhuvan University
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science and Information Technology

by
Sandeep Aryal
March, 2013

Tribhuvan University

Institute of Science and Technology

COMPARATIVE ANALYSIS OF PARTICLE SWARM OPTIMIZATION VARYING THE INERTIA FACTOR

Dissertation

Submitted to:
Central Department of Computer Science and Information Technology
Tribhuvan University
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Computer Science and Information Technology

by

Sandeep Aryal

March, 2013

Supervisor
Prof. Dr. Shashidhar Ram Joshi

Tribhuvan University

Institute of Science and Technology
Central Department of Computer Science and Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than the listed here have been used in this work.

…………………….
Sandeep Aryal

March 27, 2013

Tribhuvan University

Institute of Science and Technology
Central Department of Computer Science and Information Technology

Supervisor's Recommendation

I hereby recommend that this dissertation prepared under my supervision by **Mr. Sandeep Aryal** entitled **"Comparative Analysis of Particle Swarm Optimization varying the Inertia Factor**" be accepted as partial fulfillment of the requirements for the degree of M.Sc. in Computer Science and Information Technology. In my best knowledge this is an original work in computer science.

…………………………..
Prof. Dr. Shashidhar Ram Joshi
Department of Electronics and Computer Engineering, Institute of Engineering
Tribhuvan University
Pulchowk, Lalitpur, Nepal
(Supervisor)

Tribhuvan University

Institute of Science and Technology
Central Department of Computer Science and Information Technology

Letter of Approval

We certify that we have read this dissertation and in our opinion it is satisfactory in the scope and quality as a dissertation in the partial fulfillment for the requirement of Masters Degree in Computer Science and Information Technology.

Evaluation Committee

…………………………..	…………………………..
Prof. Dr. Shashidhar Ram Joshi Department of Electronics and Computer Engineering, Institute of Engineering, Tribhuvan University Pulchowk, Lalitpur, Nepal **(Supervisor)**	**Asst. Prof. Nawaraj Paudel** Central Department of Computer Science and Information Technology, Tribhuvan University Kirtipur, Kathmandu, Nepal **(Acting Head of Department)**
………………………	**………………………**
(External Examiner)	**(Internal Examiner)**

ACKNOWLEDGEMENT

I would like to acknowledge my respected teacher and dissertation advisor Prof. Dr. Shashidhar Ram Joshi, Department of Electronics and Computer Engineering, Institute of Engineering, Pulchowk, for giving me an opportunity to work under his supervision; providing me the guidance and support throughout this work. I wish to extend my sincere appreciation to the respected Head of the Central Department of Computer Science and Information Technology, Dr. Tanka Nath Dhamala and the Acting Head of Department Asst. Prof. Nawaraj Paudel for their kind help, encouragement and constructive suggestions.

I would also like to thank my teachers for providing me the best of academic times in the two years duration at the Central Department of Computer Science and Information Technology. My utmost respect goes to them, for, without them this work would not have been possible.

In addition, I would like to express my gratitude to the staffs at Central Department of Computer Science and Information Technology who have helped me during the period. Last but not the least, seniors and friends remain the best part of the period. They are the ones who really gave in the effort to build a really great learning environment. I thank them all.

ABSTRACT

Finding a sub-optimal solution to a difficult problem sometimes is better than finding the optimal one. It results in the reduction of cost in terms of time and feasibility. Approximation algorithms do the same thing. Among the different optimization techniques for different optimization problems, approximation algorithms help in finding approximate to optimal results. In this dissertation, an implementation of the Particle Swarm Optimization, an approximation algorithm, has been provided. Different parameters as found in the Particle Swarm Optimization have been varied. The impact of the variation in the algorithm has been studied with respect to three standard benchmark equations namely, Parabola, Rosenbrock and Griewank and statistically analyzed afterwards. The main area of this work however, goes through the variation of the Inertia factor in the algorithm. This factor has been varied with the values that go through arithmetic, geometric and harmonic sequence. The impact or the resulting effects of the variations for the benchmark equations have been provided with the statistical analysis of the results. The work then gives a suggestive approach on the selection of progression when varying Inertia factor through arithmetic, geometric and harmonic sequence in the simplest form of Particle Swarm Optimization algorithm.

Keywords: Approximation Algorithms, Swarm Intelligence, Particle Swarm Optimization, Inertia Weight, Mathematical Progressions,

For my family

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
ANSI	American National Standards Institute
AP	Arithmetic Progression
CA	Cellular Automata
D	Dimension
EA	Evolutionary Algorithms
GA	Genetic Algorithm
GP	Geometric Progression
GBEST	Global Best
HP	Harmonic Progression
LBEST	Local Best
OOP	Object Oriented Programing
PSO	Particle Swarm Optimization
SI	Swarm Intelligence
TSP	Travelling Salesman Problem

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

CHAPTER 1

1. AN OVERVIEW OF THE WORK

1.1. INTRODUCTION

Particle Swarm Optimization (PSO) is an evolutionary approach to optimization technique. It is a population based stochastic technique inspired by bird flocking and fish schooling. First developed by James Eberhart and Kennedy in 1995, the algorithm maintains a swarm of particles in a multidimensional space. Each particle in the swarm is a potential solution to a given problem [1]. These particles move according to a fitness criteria to find an optimal solution. The movement of each particle is based upon three factors: Inertia of the particle, the best position the particle has attained so far and the best position attained by the whole swarm. These factors play a crucial role in the movement of each particle in the space. Inertia represents the factor that involves continuation of motion towards the previous direction. Another factor the personal best factor influences the movement of the particle with self-cognition while the global best factor influences the movement of particle with social-cognition [2].

Since the development, PSO has been studied with many modifications — modifications primarily done to get a better convergence in a problem. There are three types of modifications in PSO: search space extension, parametric adjustment and hybridization with other techniques [3].

Parametric adjustments involve factors that play vital role in the PSO algorithm. Inertia weight [4], cognitive and social coefficients, velocity clamping, velocity constriction, velocity models, pbest and gbest modifications and topological arrangement of particles in multidimensional space are some such parameters.

The parameter, Inertia weight, has been tested and studied for different values. The testing method that has been much used is one that consists of making this value decrease over course of time with the increase in time with each iteration [4-7]. Trial and errors in its variations have helped tune this value. The change of different values for this attribute affects the algorithm in local and global searching [8]. In this context, this thesis intends to study the comparative analysis of PSO in varying the Inertia factor through geometric progression, arithmetic and harmonic progression.

1.2. Motivation

PSO algorithm although being a relatively new algorithm, has been in research with great aspirations. Numerous research on PSO today makes it one of the more sought after algorithm for modifications useful in different research areas. And it has been proven to work effectively on many real-life problems also. This dissertation, taking the effectiveness and simplicity on account, is motivated towards adding up some more information with simpler modifications on a parameter, namely, the inertia factor.
The PSO algorithm is stated in the form of equation as follows:

$$\mathbf{v_{i+1} = w \times v_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i)} \qquad \text{(Equation 1.1)}$$

$$\mathbf{x_{i+1} = x_i + v_{i+1}} \qquad \text{(Equation 1..2)}$$

where,
v_i is the current velocity of the particle in a multidimensional space.
x_i is the current position of the particle i.; a multidimensional vector quantity
w is the Inertia weight
c_1 is a cognitive coefficient,
c_2 is the social coefficient.
p_i is the personal best position of a particle i.
p_g is the global best position of the swarm.
r_1 and r_2 are random numbers.
The succeeding velocity and position are subscripted i+1.

The above algorithmic equations 1.1 and 1.2, has many parameters. Among them, c_1 represents the coefficient of self-cognition or cognitive coefficient. This coefficient affects the equation with the factor that involves the previous best position attained by any particle in the space [9]. Another coefficient, c_2, represents the factor that influences the particle towards the globally best position attained by the swarm. The Inertia parameter w is the parameter that influences the movement of any particle towards its inertial direction. The variation in this parameter thus leads to exploration — the covering of a large area in the given space and exploitation — the fine tuning of the region of optimal solution [9].

The values for w, c_1 and c_2 have been tried and tested for different values. The standard implementation [9] used values w=0.7, c_1=1.4 and c_2=1.4 for their work. These values came using the trial and error approach and could differ according to the problems and scenario of the problems defined. For the same reason, this thesis intended to get an Inertia value when the same was tested for Arithmetic, Geometric and Harmonic Progression and hence was set the problem for study. The problem further extended to the analysis under the variations and selection the proper choice of progression.

Many standard PSO implementations take the value of the Inertia parameter considering many suggestive works on the field. The lack of concrete and fixed value for this parameter [10] makes it open for comparative analysis of PSO as a whole in varying this factor. As such, for this dissertation, the Inertia coefficient was varied according to arithmetic, geometric and harmonic progression. The variation impact in the algorithm was comparatively studied.

1.3. OBJECTIVES

The aim of this work was to make a comparative analysis of the PSO algorithm in varying the Inertia parameter. Although many studies similar to this existed prior to this work, Inertia factor varied under harmonic sequence was a new study and hence the objectives were set as follows:

1. To make comparative analysis of standard PSO algorithm varying w.
2. To study the exploitation and exploration behavior of swarm in varying w.
3. To propose a better value for w in this variation.

1.4. APPLICATIONS

The PSO algorithm has many applications: human tremor analysis, computer numerically controlled milling optimization, ingredient mix optimization, reactive power and voltage control, battery pack state-of-charge estimation are some as listed in [9].

1.5. ORGANIZATION OF THE WORK

The rest of the work is basically divided into chapters. Introduction and theoretical backgrounds are discussed in the next chapter, followed by literature review, research methodology, results and analysis in the subsequent chapters. The work concludes with the discussion on the results achieved and further recommendations. Appendix is provided with the source code of the algorithm implemented within this work.

CHAPTER 2

2. THEORETICAL FOUNDATIONS ON PARTICLE SWARM OPTIMIZATION

2.1. SWARM INTELLIGENCE

Swarm Intelligence, a modern artificial intelligence discipline, is concerned with the design of multi agent systems. It is a fundamentally different design paradigm from traditional approaches to optimization. It focuses not on a single sophisticated controller that governs the global behavior, but on many unsophisticated entities that cooperate to exhibit a desired behavior. The inspiration for the principal has been taken from collective behavior of social insects such as ants, termites, bees, and wasps as well as from the behavior of other animal societies such as flocks of birds or schools of fish [2].

In such animal societies, the behavior of an individual does not mean much; however the collective behavior of each individual produces extraordinarily great results. These phenomena have made mathematicians and physical scientists to define mathematical models based on such behaviors [11]. Many such events can be seen in the insect world. Ants, termites and wasps build sophisticated nests collectively. They are not guided by a single master mind or a master plan as to how to proceed. Another such amazing event is the searching of food in bees.

Ants always find the best path from their nest to the food source, they make use of chemical pheromone trails in order to do so; bee colonies exploit richest food sources based on scouts that communicate information about the sources by means of a so-called waggle dance [11].

Swarm Intelligence is the term for the field of research which is inspired by such natural collective phenomena. Two promising areas where swarm intelligence has been in use are optimization and swarm robotics. These fields make use of information exchange in collective behavior of entities. These have been found quite successful. Some other areas where swarm intelligence has been prominent are routing and load balancing in telecommunication networks [11].

2.1.1. Swarm intelligence and its biological foundations

One may be surprised because of the discussion on biology here. One could even question on the discussion. So, why is biology here? It is because, biological phenomena has myriad of examples where successful existence of beings are found. Persistence of living beings due to biological phenomena does provide inquisitive avenues. And biology is here just to encourage and inspire computer scientists and researchers [11].

Inspirations can be taken because; the consistency of biological entities from the very beginning of time in withstanding the different changes is something great. Mathematical models can be built upon them. In fact, many such models exist and many phenomena can be explained using mathematics.

The behavioral aspects found in biology can be promising area where mathematical models can be built upon. Many instances can be found where the inquisitions in nature can be addressed only when mathematical models are built upon them. One such instance is the shape of the cells in a bee hive. Each cell in the bee hive is shaped hexagonal; why do the bees build them so? Is it because the shape can accommodate more honey? Well, this could have been an answer were it not for R.A.F de Reaumur who realized that the amount of wax needed made it so. Also, it is the consequence of the shape of each cell that the given area is divisible into equal cells. It is surprising but are the bees blindly using the highest level of mathematics? The answer to it may or may not lie within the deepest secrets of biology. However, some discussions on the event can be found in the words of Charles Darwin. Darwin is renowned for his theories. His “Origin of Species”, “Survival of the fittest” and the “Natural selection” theories have proven to be an asset in studying living science. And, for the shape of cells in the bee hive he stated that the ancestors of bees experimented with the shapes. After some experimentation the hexagonal shape gave the better result, and hence, it persisted. According to Darwin, the process of natural selection consequently led to the hexagonal shape. The shape finally persisted only after going through numerous, successive, slight modifications of simpler instincts [11].

On the contrary, however, in his book “On Growth and Form”, D' Arch Wentworth Thompson [12] dismissed the idea that natural selection alone that took toll on the shapes of the cells. He emphasized that simple physical or mathematical rules accounted for the hexagonal shape.

Thompson, being a biologist and a mathematician himself remarked that the patterns simply formed due to physical forces and no natural selection applied in it [3]. It was indeed proven to be true from the studies of formation of shapes of waxes when exposed to such environment. They shaped hexagonal. Thus, even in such complex phenomenon, no such thing as a divine entity exists. Having dismissed the idea that all the biological happenings have diving entity behind them, scientists today are finding new things from biology. The idea also puts a foundation on which one can say that the underlying principles that govern different biological happenings can be common to the physical happenings. Physical happenings can be studied with mathematics; mathematical models can be built out of biological phenomenon.

The physical study of the honeybee comb, for instance, is quite interesting on this aspect. The honeybee's comb is great for its hexagonal shaped cells; it is greater for the arrangement of brood (eggs; future larvae), pollen and nectar (future honey). Their arrangement in the comb is of a characteristic pattern. The pattern of the comb consists of three clearly distinguishable concentric regions: a central brood area, a surrounding rim of pollen and a large peripheral region of honey. This pattern is even more pronounced in the central area of the comb where a large portion of brood volume intersects. Yet another question arises, how would the humble bees know such a structure? Is it spontaneous? Do they have blueprints? The egg-laying behavior of the queen along with the movement of foragers (food searching workers) and the behavior of the nurse workers, those that feed the brood was monitored by Camazine [13]. In his study, he found that the queen laid her eggs within the central area. She laid her eggs in the cells closer to a cell containing another brood and never went more than few cell lengths of the brood-containing cell.

The workers on the other hand, were quite random at their selection of cell into which to deposit the pollen; had no preference in selecting the site. However, when removing the pollen and honey, the closest to the brood were removed with preference. This preferential removal of pollen and nectar from cells closest to cells containing brood contributed to the characteristic pattern in the honey comb. This pattern was also attributed to the queen's preference of laying eggs.

To understand such a complex process, Camazine, with the help of a simulation model modeled his observations. By constructing a simulation model based on his behavioral observations, he was able to closely follow the emergence of the pattern. Random structure of honey and pollens is developed when the workers do not have any preference in depositing the pollens. The characteristic structure is defined because of the movement of honey towards the brood area and the preference of the queen in laying eggs. Camazine could find this fact only using computer simulation.

With no tools like simulations or mathematics, it is impossible to translate individual behavior into collective behavior. These tools, available outside the biological studies, describe the collective nature of emergence of characters. Each individual corresponds and represents an entity in the tools and a being in the biology respectively.

Prior to this, it would be impossible to translate individual behavior into collective behavior since tools such as simulations or mathematics were not used. However, tools available today — tools that are outside biology — have been effective in dealing on behavioral attributes with the view that individual interactions bring about collective behavior.

In saying so however, the generalization in this context is far more complex in case of biology where natural selection acting upon them cannot be ignored. Natural selection and divine entities cannot be ignored altogether. Natural phenomena are of stochastic behavior.

On the contrary, if a mathematical model for some system captures the behavior of some others, we can always produce an analogy and talk about their similarities. Such similarities could be generalized. They may lead to some collective study.

2.1.2. Decentralized Decision Making

In the natural phenomena such as bees, wasps searching for food or for a new home; decision is not taken solely. In fact, a member only recommends his or her status if it is better than others. If it is not, then he or she moves towards a better positioned member. Bees, wasps and some other animal colonies have members who transfer information in between. Their solitary existence is not counted in the group making their sole decision less influential. Insect colonies need to make collective decisions, for example where to forage, which nest to move to, when to reproduce and how to do the division of labor. It is also known that group-level decisions are the results of the individual insects acting mainly on local information obtained from interactions with their peers and their immediate environment [14]. In other words, decision making in insect societies is decentralized. This has been seen and proven by experimental studies on foraging and house moving behavior of ants and bees.

2.2. Swarm Intelligence in Optimization

2.2.1. Introduction

Swarm intelligence is based on the idea of collective behavior shown by insects such as ants, termites, bees, and wasps, and other social animal groups such as flocks of birds or schools of fish. It is concerned in creating multi-agent system based on these and it is considered to be a discipline from artificial intelligence.

Insect colonies have produced curiosity for researchers for a long period of time. The mechanisms that govern their working have been a mystery for many years from now. The colonies produce a very sophisticated output although each individual may not seem to be capable. Complex tasks in get achieved with remarkable sophistication in cooperation. Simple actions and co-ordination among the individuals bring about collective behavior. Many aspects of achievement in such a feat happen due to coordinated approach. As an example, pieces of leaves are brought by leaf cutter ants to the nest. They use the pieces in order to grow fungi on them. The fungus is used as a food source for their larvae. Weaver workers build chains with their bodies when they need to cross gaps between two leaves. These leaves get connected with the silk produced from a larva. The larva is brought by a worker and the edges of leaves get stitched. The recruitment of other colony members for prey retrieval can be taken as another example. Sophisticated nests are built by termites and wasps. Bees and ants adapt themselves in their environment pretty much good. These could be few other examples to talk about.

Beni first used the term swarm intelligence in the context of cellular robotic systems. In them simple agents organized themselves through nearest-neighbor interaction [11]. Meanwhile, today, the term swarm intelligence is used for a much broader research field. The methods used in swarm intelligence have been very successful in the area of optimization. Optimization in itself is of great importance for industry and science. This work primarily focuses on Swarm intelligence and its variant — The Particle Swarm Optimization.

Optimization problems can be found in industries and in scientific researches. Practical problems such as train scheduling, timetabling, shape optimization, telecommunication network designs and other problems found in computational biology all have optimization issues. These problems need to address the optimization of their systems. Also, in research problems such as a well-known traveling salesman problem (TSP), optimization has a major role. This problem has the scenario where a traveling salesman needs to pass through a number of cities. The goal of the traveling salesman is to traverse these cities. The traversal however, contains the criteria that he finds the shortest possible solution visiting each city only once. Another example that requires optimization is the problem of protein folding. This problem needs to address the finding of a functional shape or conformation of a protein in two or three-dimensional space; the problem itself is considered to be one of the most challenging problems in computational biology, molecular biology, biochemistry, and physics. This problem and the TSP belong to an important class of optimization problems better known as combinatorial optimization (CO).

Any optimization problem P can be defined in terms of a triple (S, Ω, f), where

1. S is the search space defined over a finite set of decision variables Xi, i = 1, ... , n. In the case where these variables have discrete domains the optimization is called discrete optimization (or combinatorial optimization), and in the case of continuous domains P, it is called a continuous optimization problem. Problems with mixed variables also exist. Constraints among the variables constitute a set called Ω.

2. $f : S \rightarrow R+$ is the objective function that assigns a positive cost value to R each element (or solution) of S.

The goal is to find a solution $s \in S$ such that $f(s') \leq f(s')$, $\forall\ s' \in S$ (in case one wants to minimize the objective function), or $f(s) \geq f(s')$, $\forall\ s' \in S$ (in case the objective function must be maximized). The target often is to optimize several objective functions at the same time in cases of real-life problems. Such forms of optimization is termed multi objective optimization [11]

Many algorithms to tackle these problems have been developed because of the practical importance of optimization problems. In the context of combinatorial optimization (CO), either complete or approximate classification of algorithms can be found. Complete algorithms are guaranteed to find for every finite size instance of a CO problem an optimal solution in bounded time. However, for CO problems that are NP-hard, no polynomial time algorithm exists, assuming that $P \neq NP$. Therefore, exponential amount of computation time might be needed in the worst case for complete methods. In practice this could not be a feasible option. In approximate methods such as SI-based algorithms there is a trade-off between finding guaranteed optimal solutions and finding significantly reduced optimal solutions in order to save the amount of time. The compromise in the guaranteed solution to save the computation time has had a great reputation in researches. Such algorithms do not deviate much from the optimal solution and are much preferred. The ease in implementing the approximate methods stands as another major advantage to its preference over classical gradient-based approaches.

Since, these algorithms do not require gradient information; it is easier for optimization problems where the objective function is only implicitly provided meaning that the objective function values are obtained by simulation or in cases where differentiation of objective functions is impossible.

Ant colony optimization (ACO) and the particle swarm optimization (PSO) are two of the most notable swarm intelligence methods for obtaining approximate solutions to optimization problems in a reasonable amount of computation time.

2.2.2. Ant Colony Optimization

First among the techniques for approximate optimization inspired by Swarm Intelligence is the Ant colony optimization (ACO). It is a technique inspired by the foraging behavior of ant colonies. The foraging behavior of ants and their communication system in between the process is due to chemical pheromone trails. This indirect communication among the ants however, always finds them the short paths between the food source and their nests. This astonishing feature of real ant colonies is exploited in ACO algorithms and is used in discrete optimization problems. From the operations research (OR) perspective, ACO can be categorized to the metaheuristics algorithms class. Other algorithms, such as evolutionary computation, iterated local search, simulated annealing, and tabu search in addition to ACO are often regarded as metaheuristics.

The Origins of Ant Colony Optimization

Ant Colony Optimization was first introduced by Marco Dorigo and colleagues in the early 1990s. Inspiration for the development of these algorithms came with the observation of ant colonies. The focus of each single ant in the ant colony is governed by the goal directed towards the survival of the colony. Ants being social insects are less focused on sole development and survival. ACO is inspired by the foraging behavior of ants and in particular how ants can find the shortest path between their nests and the food source. When searching for food, ants initially explore the area surrounding their nest in a random manner. However, the ants leave a chemical pheromone trail on the ground while they move. Ants can smell pheromone. Their choice of path and their tendency toward choosing their way is very much affected by the pheromone concentrations on the way. When an ant finds a food source it takes some portion of it and as it returns it leaves on the ground pheromone. The pheromone concentration depends upon the quantity and the quality of the food source. This pheromone trail guides other ants to the food source. This process — better known as stigmergy — enables them to find shortest paths between their nest and food sources. This amazing phenomenon, which has had so much of curiosity to researchers, has now been adapted with mathematical models and approaches [11].

2.2.3. Particle Swarm Optimization

Particle swarm optimization (PSO) optimization technique is population based, stochastic technique for optimization modeled on the social behaviors observed in animals or insects, e.g., bird flocking, fish schooling, and animal herding. PSO was originally proposed by James Kennedy and Russell Eberhart in 1995 [2]. The conceiving of the theory has gained so much popularity among researchers and practitioners that PSO has been taken as a robust and efficient technique for solving difficult optimization problems.

Each individual particle in a PSO is a potential solution. Each particle moves through a problem search space seeking an optimal or better found solution; broadcasts its current position to neighboring particles seeking an optimal or better found solution. This reference is taken by all, compared and then appropriate movement is made so that each particle moves towards the best position found. The movement however is not only affected by this factor but variations exist and will be discussed later on. This progresses further and on each such iteration the swarm concentrates more and more on the area of the search space that contains high-quality solutions.

PSO is very close to models that characterize artificial life. The rules that govern the emergent behaviors of bird flocking for instance, is greatly due to the local interactions that take place in between the group members. Such studies laid the foundation for the subsequent development of PSO; its use in optimization. In fact, PSO is in some ways very similar to cellular automata (CA). CA are used for generating recurring patterns based on very simple rules. John Conway's Game of Life, for instance uses three simple rules:

1. Each individual cell is updated in parallel

2. The value of each new cell depends only on the old values of the cell and its surrounding cells and
3. The updating criterion for all cells is the same.

The analogy between the PSO and CA exists; the analogy being the updating of cells in all dimensions simultaneously in the former case. James Kennedy and Russel Eberhart, first coined the name Particle Swarm Optimization (PSO); and the first ever invention of the original PSO is due to them. Their first intention was to model the movements of flocks of birds and schools of fish. As their model further developed and evolved to handle optimization, swarms of mosquitoes like displays were seen in their visual plots. Because motion was concerned in the study, the term particle was used. The term particle actually seemed more than appropriate later on in the context [11].

2.3. OPTIMIZATION

Optimization means the selection of best alternative among many alternatives. The selection process may go through certain criteria — the criteria set is termed “constraints” in the optimization theory — that may be required to be fulfilled. If the selected alternative fulfills all the criteria defined over the problem, then the alternative is a feasible solution. Feasible solutions are not always unique. In fact, for any optimization process there are usually many of them. Among the feasible solutions, picking up the best alternative is optimization. The process consists of either maximizing or minimizing a real function — function of a mathematical model or a function defining a physical model — by choosing input values from within an allowed set or the domain of the function. In a general sense an optimization process consists of picking the best solution from an allowed domain of possible values, possibly constituting a function, and following some constraints that should be met. A large area of applied mathematics revolves around the generalization of optimization process.

2.3.1. An Optimization Problem

An Optimization problem is the problem of finding an optimal solution where, an optimal solution is the one that is the best among all feasible solutions to a given problem. However, the best solution or the globally optimal solution is not always the preferred one for a problem. It depends upon the nature of the problem and the trade-offs between finding the global optimal and finding a less optimal or suboptimal solution. Sometimes a suboptimal or a locally optimal solution is preferred over the global solution. This is usually described as local optimization. If a problem though, requires finding the globally optimal solution then it is called global optimization.

An optimization process is always preceded by a modeling phase. In the modeling phase actual problem is modeled mathematically. Constraints involved are also incorporated during this process. Numerical variables are taken into consideration when modeling the actual problem solutions. These variables are then used to map the actual problem into a function. Such function is called an objective function. Therefore, optimizing this objective function is related to a real problem optimal solution. Minimization or maximization of the objective function gives a map to the optimal solution to the real problem. Since, minimization or maximization can equivalently defined by changing their signs, minimization is taken as a general case in this thesis.

The objective function is always associated with a domain. The domain consists of allowed set of values for the evaluation of the function. These values are probable solutions to the function and these too need to conform to some defined set of constraints. In other words, problem constraints delimit the domain. Constraints also need to be quantified properly in a mathematical model. The variables that define the mathematical model ultimately reflecting the real physical problem are bound to these set of constraints; the constraints themselves reflecting the boundary of the real problem. In the simplest of cases, the variables are only bounded by the constraints, however in harder problems, the variables maintain complex relationships among themselves rendering the minimization process rather difficult.

Some problems are easier than others to minimize. For instance, if a function is differentiable and has a relatively simpler form, then zeros of its gradients can be used as minimizers. Some are not so simple. They simply don’t fit into the mathematical assumptions. Algorithms that approximate the optimal solution become inevitable in the latter cases. These algorithms work iteratively producing a sequence of search points. The sequence provides some subsequences that ultimately converge to actual minimizers.

For decades, optimization has been an active research field. The technological advances and booming of different possibilities have not only brought about a revolution in the world but they have also brought forward different and more complex optimization problems. These have urged for different approaches in optimization techniques and triggered for more advances in algorithms. There has been a sophisticated advancement in algorithm technology. Nevertheless, real world optimization suffers from the following problems: [12]

1. Difficulty in distinguishing global from local solutions
2. Presence of noise in solution evaluation.
3. Exponential growth of the search space with the problem's dimension.
4. Problems constraints associated difficulties.

Mathematical modeling and the nature of mathematical problems are not similar among themselves. So is the case when these models define real life problems. Some are linear, others nonlinear. Convexity, differentiability, continuity, function evaluation accuracy are some other factors that define the mathematical modeling category. These cannot be generalized easily and need specific treatment for optimization. Specializing in the optimization process means specialization of evaluation and the specialization of algorithm. An algorithm to a specific problem of optimization may not work for others. However, some of the common properties that can be defined over many models can be tackled by using some algorithmic solution that can be inherited by all. Problems on stochasticity, for instance, could be a common one that can be used by many algorithms commonly.

Algorithms are easily available today. For most of the problems, an off-the-shelf algorithm can be used. Nevertheless, different instances of the same problem may have different computational requirements, leaving space for innovation, innovation that may develop into a new algorithm or modification of the existing one. The result is that there will always be sophistication of idea and development of algorithms.

2.3.2. Types of optimization problems

An optimization problem can be defined differently according to the problem type. In general, any function f:X→Y, defined over a domain , X, also called the search space, and with range, Y, can be optimized given the total ordering relation over Y. The most basic optimization consists of minimization of functions whose domain is a subset of n-dimensional Euclidean space, $\mathbf{R}^n$, and their range is a subset of real numbers. The problem may have constraints also. Thus, the minimization problem can formally be described as:

$$\min_{x \in A} f(x), \text{ subject to } C_i\ (x) \leq 0, i = 1, 2, \ldots, k$$

where, $A \subseteq \mathbf{R}^n$, is a subset of n-dimensional Euclidean space; $Y \subseteq \mathbf{R}$, is a subset of all real numbers; and, k, is the number of constraints.

Now, it can clearly be said that x is a feasible solution to the function f(x) only if it satisfies the constraints function. When constraints are present, it is a constrained optimization problem and when they are absent it is called an unconstrained problem.

Based on the properties of objective function, its domain, and the forms of the constraints optimization problems can be further divided. Some of them are:

1. Linear optimization (or linear programming): cases where objective function and constraints are linear.
2. Nonlinear optimization (or nonlinear programming): cases where at least one non-linear function is involved.
3. Convex optimization: problems with convex objective functions and convex feasible sets.
4. Quadratic optimization (or quadratic programming): minimization of quadratic objective functions and linear constraints.
5. Stochastic optimization: minimization of presence of randomness usually based upon probabilistic selection of problem variables and parameters and statistical distributions.

In the usual cases, optimization process involves single objective function that does not change throughout the optimization process. However, many other problems exist where the objective function changes over time or conditions and sometimes constitute multiple objective functions. These variations further divide the optimization problems into the followings:

1. Dynamic optimization: minimization of time-varying objective functions.
2. Multiobjective optimization: concurrent minimization of two or more objective functions.

Yet another classification is based upon the nature of search space of the variables:

1. Discrete optimization: variables of the objective function assume discrete values. When the variables take integer values, it is specially called Integer Optimization.
2. Continuous optimization: all variables involved assume real values.
3. Mixed integer optimization: both integer and real values get involved.

There are plenty of methods available to solve the given types of optimization problems. The abovementioned problems however, are based on strong mathematical assumptions. Real-world applications do not have strong mathematical assumptions. They cannot be accurately defined by any mathematical model and this holds true for most of the real-life problems. [3]

Even mathematical models do not guarantee for all the above mentioned problems unless the problems themselves comply by the mathematical process requirements. For instance, in case of nonlinear optimization differentiability of objective function must hold which may not be possible at times. Even if the mathematical model does provide the optimal solution, they may not be satisfactory. So, to help optimize real world problems without taking much consideration on the formation of internal structure of the problem or to optimize in case where information about the underlying problem is least, newer concepts are being developed. Among such optimization algorithm techniques, one is particle swarm optimization.

There are two categories of optimization algorithms: deterministic and stochastic algorithms. Deterministic type of algorithms can be expected to produce the same steps in similar types of problems and the nature of exactness over the steps becomes their prime feature. Stochastic algorithms, on the contrary, hardly produce identical values in their steps even if performed in the same types of problems.

Deterministic approaches require strong mathematical assumptions and only then, these algorithms can be used. Stochastic methods include random search, probabilistic models of objective function based methods and clustering. Most of these models iterate over the steps in order to refine the possible solution at each step. Since iteration over the search space is used, strong mathematical assumption is not required in such algorithms, albeit such assumption could always help reduce the number of required steps.

Although many different classification of optimizations may have been defined over the period of time, there are certain classifications that need to be mentioned in order to make some fundamental basis for the main approach in this thesis— The Particle Swarm Optimization.

One such is the direct search algorithm where the algorithm heavily relies upon the evaluation of the objective function over some values. Generalized descent methods require first and second order differentiation in order to find the region of minima. Clustering methods aims at reducing the limitations of the generalized descent approach by producing a cluster around a region so as to find the local minima. Then, single local search can be conducted per cluster.

Random search algorithms are based on probability distributions. The distributions help produce sample of search points. In pure random search, every iteration produces a new set of sample points. The output accuracy increases as the sampling size approaches infinity. In practice, pure random search is considered to be the worst acceptable performance for an algorithm. This is because pure random search requires vast number of function evaluations even for small problems. However, its variants have been seen to be more promising.

Heuristic and metaheuristic approaches have been developed that are inspired from natural phenomena. Some of them simulate fundamental elements and procedures that produce evolution and intelligence in nature. They are very much the recent fields of study. Algorithms such as the evolutionary computation and swarm intelligence for instance, are the contemporary stochastic algorithms in the research field. These algorithms hence, are the prominent part for this dissertation.

Another algorithm called the indirect search uses either Bayesian model to build the model of objective function by using local information or uses polynomial approximation to fit its level sets. Though these sound very interesting theoretically, they have lots of difficulty when practically implemented.

2.4. EVOLUTIONARY COMPUTATION AND ITS DEVELOPMENT

The term evolutionary computation is a term for optimization technique that basically lies in the heuristic algorithms category. These however, are different because, they use stochasticity, adaptation and learning process during their implementation. The learning and adaptation phase always have some part related to previous steps. Each former step helps lead the evolution in the steps that follows.

Darwinism is sometimes related to evolutionary approach to optimization. Population of potential solutions iterates throughout the run producing descendants more and more of a potential to be optimal. During the process, many approaches handle the evolution process. Selection, mutation and crossover are some of the techniques that help evolve the population. Since these processes are the same as described by Darwin in his natural selection theory, evolutionary optimization approaches are related analogously by some.

Some though, do not believe firmly on the relation. Although Darwin's theory had been compared and used from as early as the mid-50's in the field of machine learning, it was until the 90's that the term of "evolutionary computation" was used as a new and promising research field.

Evolutionary programming, evolution strategy and genetic algorithms are the different fields of research that have been followed in the US and in Europe in the recent past. This has been proven by the huge number of researches, published books, journals and conferences organized all over the world under these topics. Brief round-up of them is discussed further.

2.4.1. Evolutionary Programming

Evolutionary programming has evolved much from its early invention, towards the 80's and now. The traveling salesman problem, neural network training, scheduling, and continuous optimization are just some problems addressed by it. It has huge amount of work to be explored upon and its relative genetic programming too has many areas to be researched upon.

2.4.2. Genetic Algorithms

Genetic algorithms have been the most popular heuristics for global optimization today. They use population of potential candidate solutions, just like other variations of evolutionary algorithms. Selection, crossover and mutation approaches revolve the genetic algorithms. The algorithm terminates as soon as the desirable descendent is found or when the computation limit exhausts.

2.4.3. Evolution Strategies

Another approach to evolutionary optimization that lacks the crossover feature is evolution strategies that constitute another interesting field. Many variations are found for evolution strategies approach [13]. This area of research is beyond the scope of this dissertation and will not be discussed further.

2.4.4. Swarm Intelligence

Swarm Intelligence is a modern artificial intelligence discipline that is concerned with the design of multi agent systems with applications, e.g. robotics and optimization. It is a fundamentally different design paradigm from traditional approaches to optimization. It is a paradigm that focuses not on a single sophisticated controller that governs the global behavior, but on many unsophisticated entities that cooperate to exhibit a desired behavior. The chief source of inspiration to swarm intelligence is the collective behavior of social insects such as ants, termites, bees, and wasps. However, behavior of other animal societies such as flocks of birds or schools of fish has also been taken as inspirational source. In such animal societies, the behavior of an individual does not mean much. On the contrary, collective behavior of each individual produces extraordinarily great results. Models based on such phenomenon have been found to be utilized by many scientists and mathematicians. Swarm Intelligence has been growing as one of the promising sectors of research.

Insects such as ants, termites and wasps build sophisticated nests collectively. When studied collectively, these insects show amazing ability — ability to work as a team. Moreover, it is even more interesting fact that these are not guided by a single master mind or a master plan as to how to proceed in building nests. And they build their nests to their best fit. Such amazing phenomena have driven scientists and researchers to involve in their social behavior study. Swarm Intelligence is the term for the field of such research that is inspired by such natural collective culture. Although relatively new, two promising areas where swarm intelligence has been in use are optimization and swarm robotics. These fields make use of information exchange in collective behavior of entities. These have been found quite successful. Some of the other areas where swarm intelligence has been prominent are routing and load balancing in telecommunication networks.

2.5. PARTICLE SWARM OPTIMIZATION ALGORITHM

PSO is mainly based on the movement of particles towards a solution. Each particle that exists in the algorithm is a solution point. Each such particle moves in order to refine its state to be in a better position. Each of the particles refines its position over every iteration. They do so until certain goal criterion is met. In this process, the velocity of each particle is modified iteratively taking into consideration its personal best position (i.e., the best position found out by the particle so far), and the best position found by particles in its neighborhood. The result is that each particle searches around its best position as well as the best position attained by its neighborhood.

To discuss further, the terms v_i will be used to denote the velocity of the i^{th} particle in the swarm, x_i to denote its position, p_i to denote the personal best position and p_g the best position found by particles in its neighborhood. In the original PSO algorithm [2], v_i and x_i, for i = 1, … , n, are updated according to the following two equations.

$$v_i \leftarrow v_i + \varphi_1 \times (p_i - x_i) + \varphi_2 \times (p_g - x_i) \qquad \text{(Equation 2.1)}$$

$$x_i \leftarrow x_i + v_i \qquad \text{(Equation 2.2)}$$

where $\varphi_1 = c_1 R_1$ and $\varphi_2 = c_2 R_2$.

R_1 and R_2 — two separate functions each returning a vector comprising random values uniformly generated in the range [0, 1].

c_1 and c_2 — acceleration coefficients.

The symbol × denotes point wise vector multiplication.

Equation 2.1 shows that the velocity term v_i of a particle is determined by three parts, v_i, the "momentum", φ_1, the "cognitive", and φ_2, the "social" part. The "momentum" term — as in laws of motion — v_i, represents effect of the previous velocity term used to carry the particle in the direction it has traveled so far; the "cognitive" part, $\varphi_1 \times (p_i - x_i)$, represents the historical effect or success that sets the tendency of the particle to return to the best position it has visited so far; the "social" part, $\varphi_2 \times (p_g - x_i)$, represents the tendency of the particle to be attracted towards the position of the best position found by the entire swarm.

The best position attained by particles in the neighborhood of the i^{th} particle is the position p_g in the "social" portion. This factor directs the swarm towards the global optimal position. If this factor is greater, the flow is influenced towards the globally optimal point. Such flow helps in covering greater area. The flow of information in between the particles however, can be controlled using different neighborhood topologies. Information propagation can thus vary with topology. Ring topology, Von Neumann topology are some of the examples of neighborhood topologies. In the case of complex problems, constricted information propagations have been found to be better; they utilize small neighborhood topologies whereas, larger neighborhoods generally perform better on simpler problems. Discussions on the topology constitute a greater research area and require much more reading. It has not been discussed here as it is beyond the scope of this work.

Generally speaking, a PSO implementation that chooses p_g from within a restricted local neighborhood is referred to as lbest PSO, whereas choosing p_g without any restriction (hence from the entire swarm) results in a gbest PSO.

Basic PSO algorithm is quite simple and easy enough for implementation. Algorithm 1 summarizes a basic PSO algorithm.

Algorithm 1 The PSO algorithm, assuming maximization

```
Randomly generate an initial swarm
repeat
    for each particle i do
        if $f(x_i) > f(p_i)$ then $p_i \leftarrow x_i$
            $p_g = \max(p_{neighbours})$
            Update velocity (see Equation. 2.1)
            Update position (see Equation. 2.2)
    end for
until termination criterion is met (fitness value sets the termination criteria for this work)
```

CHAPTER 3

3. RESEARCH METHODOLOGY

The methodology used for the above proposed work involves the following research works and studies. The problem taken in the work is purely of stochastic nature and therefore, quantitative method is not approached. The findings and analysis however have been transformed to easily readable formats such as tables and graphs.

3.1. LITERATURE REVIEW

Russell Eberhart and James Kennedy in their paper proposed the theory for optimization that was based on bird flocking and fish schools. The theory was a new method for optimizing and was called the Particle Swarm Optimization. A quite short, superficial summary of their paper “A New Optimizer Using Particle Swarm Theory” can be put in a paragraph; although by no means is this version complete:

In the paper, Particle Swarm methodology has been used for optimization of nonlinear functions. Two paradigms have been implemented and compared that include the then recently developed locally oriented paradigm. Applications including neural network training and robot task learning have been proposed. The paper also provides the benchmark testing of both paradigms. [1]

This paper provides an idea on particle swarm optimization concept. In order to help give the idea, the concept has been implemented using two paradigms: globally oriented (GBEST), and locally oriented (LBEST). This has been followed by the results obtained from applications and tests upon which both the methodologies have been shown to work successfully. [1]

Since its development, PSO has been accepted by many scientists and researches with great reception towards its ability. Works, researches have flourished on the topic because of the simplicity in the algorithm. Many new problems have been found to be equally solvable using PSO and in some cases PSO has been proven to work better. Papers and reading materials such as journal articles, research papers and implementations mainly come from the Internet open sources and from books on the implementations.

3.1.1. Particle Swarm Optimization: Some Variations

Since the beginning, PSO has been tried and tested by many. The trials that PSO has gone through consist of the variations in its implementations. PSO parameters are the chief ones that have been varied. Some of the variations and the effects of the variations are given below.

The Velocity Variation

The velocity parameter, v has been varied with many different results. While the detailed discussion on this variation is quiet interesting, the details go beyond the scope of this work. However, the chief possibility on the variation of v and its impact on the overall algorithm cannot be surpassed without discussion. To summarize the impact of v varying, the parameter v always has a maximum limit to it. Letting v unlimited can make the particles to leave the search space. So, the v limited to maximum value V_{max}, has the beneficial effect of preventing the explosion (going beyond the search space) of the swarm. It scales the exploration of the particle's swarm. However, the value for V_{max} requires the understanding of the problem. For some problems V_{max} could have larger values and when provided with less the particles may get stuck in local optimum. For others it may require V_{max} to be small, giving it a larger value could disturb the fine tuning of the solution. [15]

The Control Parameter Variation

The control parameter, φ, combined form of φ_1 and φ_2 for this discussion, can also have different effects when varied. It is a parameter that takes on random values in the algorithm. The randomness in the movement of particles is provided by this parameter. Varying this parameter thus produces oscillatory movements of particles. Once again, it will be beyond the scope of this work to discuss thoroughly on the variation and the effects of this parameter. However, most prominent effect and variation cannot be left without being discussed. The value of φ is preferred by many to be between 0 and 4. This has been preferred on the basis of different tested implementations of different researchers. However, when varied beyond the upper limit, φ produces the effect of the swarm explosion. [10]

Inertia Weight

Inertia weight factor is one of the other variations of the standard PSO algorithm. This factor influences the Inertia of each particle and thus has the ability of altering the flow of whole swarm. The Inertia Factor modifies the standard PSO in a subtle way. The standard equation has now w added to it and is written as:

$$v_i \leftarrow w \times v_i + \varphi_1 \times (p_i - x_i) + \varphi_2 \times (p_g - x_i) \quad \text{(Equation 3.1)}$$

$$x_i \leftarrow x_i + v_i \quad \text{(Equation 3.2)}$$

Since this dissertation is primarily concerned with the study of the variation of this factor, much of the discussions that follow will be focused on this variation.

As mentioned previously, the Particle Swarm Optimization approach to solving optimization involves many parameters — parameters that can be easily varied and studied. That is to say that the algorithm consists of parameters each of which has their own significance. The parameters when modified have some diverse effects. The diversity of effects on the modification of the PSO parameters has been in research for considerable period. There have been many interesting results and much more interesting interpretations.

Among the variations of parameters in PSO, the variation of w, or the Inertia weight, has the effect that a particle moves in the direction where it was previously headed. Inertia is the resistance factor of the particle such that the particle does not change its previously headed direction. An analogy can be made with the inertia as found in the laws of motion. With so much discussion on w, the PSO equation can be written in the form:

$$v_{i+1} = w \times v_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i) \quad \text{(Equation 3.3)}$$

Equation 3.3 shows that the velocity v, a vector quantity, has in its composition the w parameter. The variation of w directly influences the previous velocity composition in the equation, thus affecting the movement of each particle in each dimension. The parameter w has a significant effect when it comes to the exploration and exploitation of the search space. Exploration of the swarm space means the searching for solution in broad area while exploitation means the fine tuning of a region in order to achieve precision.

3.1.2. Benchmark Set:

Particle Swarm Optimization algorithm is an algorithm that performs functional evaluations at feasible points to a given problem. Although it has been used for various other purposes, it is easier to study its performance with the help of some benchmark equations. Benchmark equations are there to test the algorithm. There are many such benchmark equations available. Among them three of the known problems have been taken as benchmark equations for this work. The complexity and the effectiveness in using the equations to test the implemented PSO will be discussed later on. However, the nature and properties of the test benchmark equations used in the work need to be discussed here.

Many equations such as Sphere, Griewank, Rosenbrock, Ackley, Rastrigin etc., can be found in standard benchmark suites such as the De Jong's test suites [16]. The proposed work however intends to limit itself to only three of the standard benchmark functions: Sphere, Griewank, Rosenbrock equations.

Function Name	Objective Functions	Search Space	Optimal Function Value
Sphere/Parabola	$\mathrm{Min} f(x) = \sum_{i=1}^{n} x_i^2$	$-5.12 <= x_i <= 5.12$	0
Griewank	$\mathrm{Min} f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$-600 <= x_i <= 600$	0
Rosenbrock	$\mathrm{Min} f(x) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right]$	$-5.12 <= x_i <= 5.12$	0

Table 3.1 Benchmarks Equations

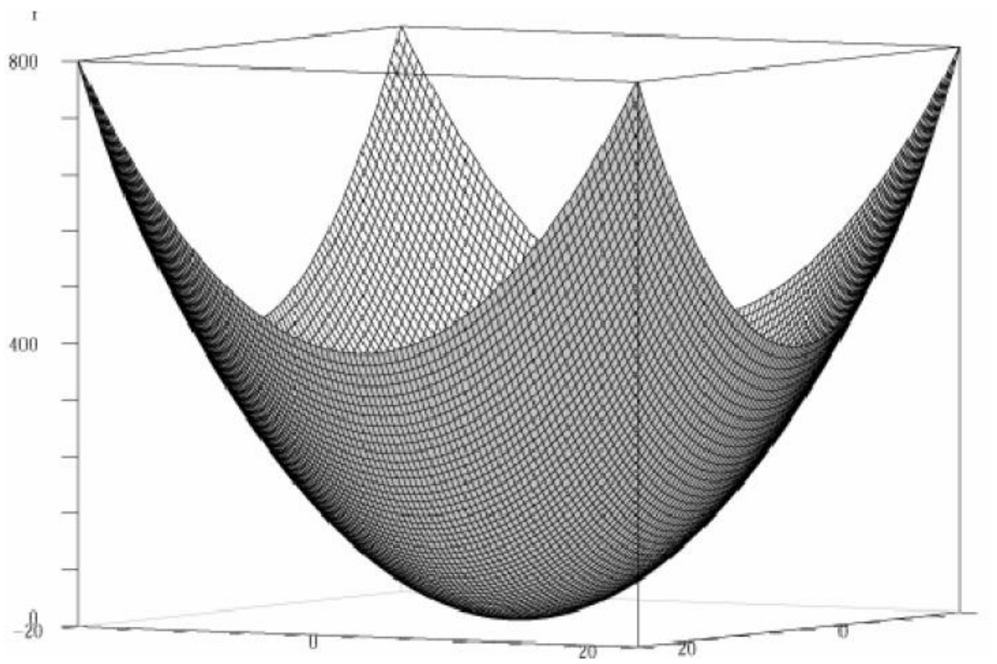

Figure 3.1 Parabola Function

Sphere or Parabola function has only one minimum. PSO might not be the best alternative to solve such problem since, PSO is stochastic in nature. Other methods such as gradient descent can be applied. [10]

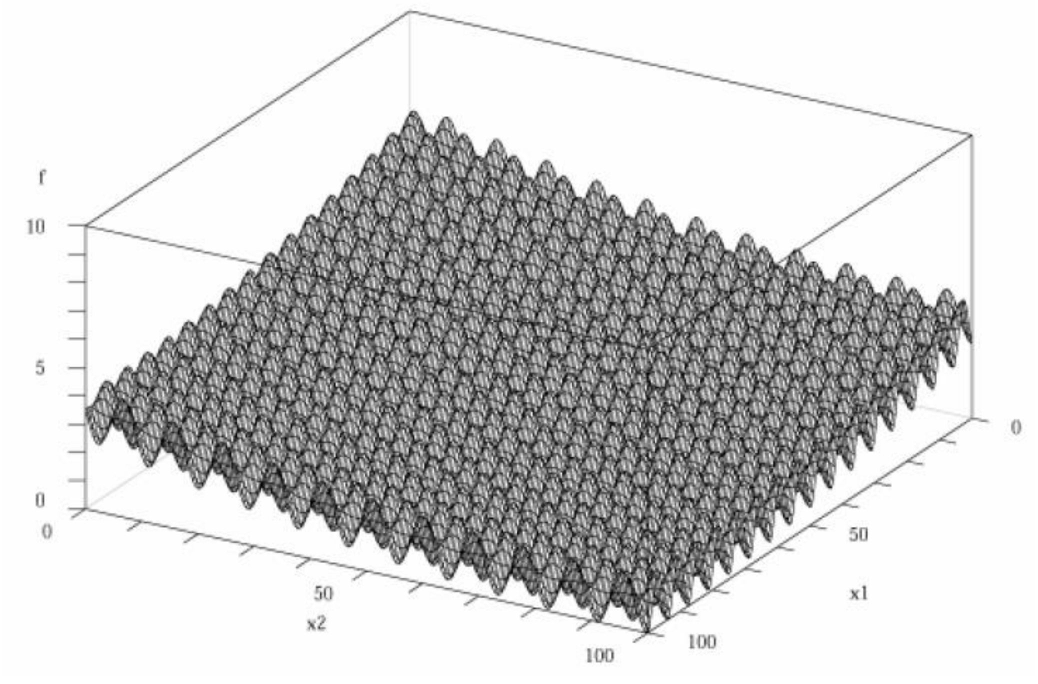

Figure 3.2 Griewank Function

It has many closely located local minima and these local minima surround the optimal solution. Almost indistinguishable global minima scenario is found making it more difficult to keep track of. However, the closely located local minima can be escaped rather easily because of their compact location.[10]

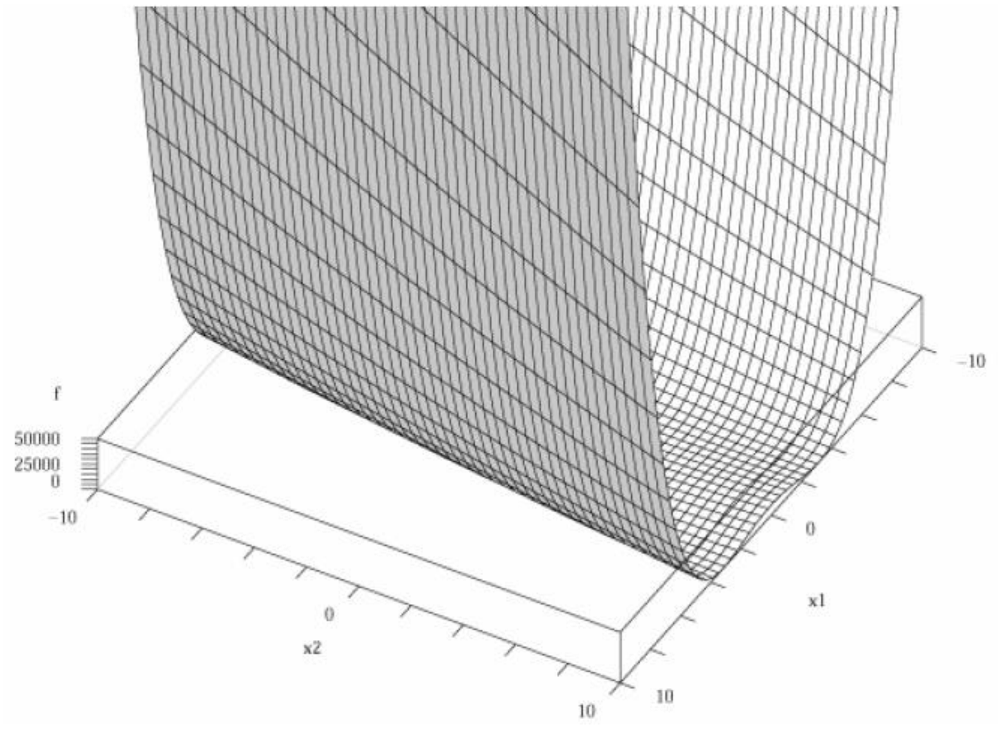

Figure 3.3 Rosenbrock Function

Rosenbrock Function has hard to notice global minimum. Majority of algorithms find it difficult to find the global optimal solution. PSO in its basic form is no different for this equation [10].

3.1.3. Mathematical Progressions

Mathematical progressions are sequences of numbers that are generated following some inherent rules. Three most basic mathematical progressions are arithmetic, geometric and harmonic progressions. These progressions are defined here since they also form an important part in this work. Variation of the w factor is done using these progressions and comprises a major part of this thesis.

Arithmetic Progression

An arithmetic progression (AP) is a progression of numbers where the difference between each consecutive term is constant. 2, 6, 10, 14 … is in AP because each consecutive term of the progression differs by a constant factor of 4. Similarly, -9, -12, -15 … is also in AP since the terms differ by constant factor of -3.

Given an initial term *a,* the common difference factor *d*, the nth term of the arithmetic sequence is given by

$$a_n = a_1 + (n-1)d$$

And, in general

$$a_n = a_m + (n-m)d$$

The sum of a finite arithmetic progression — the one with finite number of terms — is called an arithmetic series.

The nature of arithmetic progression depends on the common difference *d*. If the common difference is:

Positive — terms will grow positively towards infinitely.

Negative — terms will grow negatively towards infinitely.

Geometric progression

A geometric progression (GP), also known as a geometric sequence, is another progression where each term after the first increases or decreases by multiplication of a common number other than zero. This common number is called common ratio. For example, the sequence 1, 3, 9, 12, ... is a geometric progression with common ratio 3. Similarly, -2, -1, -0.5, -0.25 … is again in GP and has the common ratio 0.5.

Given an initial term *a,* the common ratio *r,* the nth term of GP is given by $a_n = ar^{n-1}$

The value of common ratio determines the behavior of GP. If the common ratio is:

1. Positive, terms will not change their sign.
2. Negative, the terms will have alternate positive and negative signs.
3. Greater than 1, the terms will exponentially grow towards positive or negative (depending on the sign of the initial term).
4. 1, the progression is a constant sequence.
5. Between −1 and 1 but not zero, the progression will decay exponentially towards zero.
6. −1, the progression is an alternating sequence
7. Less than −1, for the absolute values progression grows exponentially towards positive and negative infinity (sign alters in between).

Harmonic Progression

Harmonic Progression (HP) is the progression formed when each of the terms found in AP is replaced by their reciprocals. In other words, when a, a + d, a + 2d, … a + (n-1)d is in AP, 1/a, 1/(a+d), … a/(a+(n-1)d) is in HP.

The sum of the terms in HP leads to Harmonic Series. The series diverges.

AP, GP and HP are used for the variation of the Inertia Factor (w), in the basic PSO algorithm for this work. These progressions are used in order to study the variation pattern of the algorithm. The resulting impact from these variations will then be compared and analyzed later.

CHAPTER 4

4. IMPLEMENTATION

The PSO algorithm is a short and simple algorithm with few numbers of lines only. The implementation of this algorithm therefore is not so difficult as compared to other computer science optimization algorithms. It is actually the simplicity and the potential of the algorithm that has attracted researchers towards using this algorithm. This dissertation has a working code of the PSO algorithm written in Python (version 2.6). The code can be run in any platform that supports Python.

4.1. PSO ALGORITHM REVISITED

The variables and mathematical symbols used in the algorithm have the same meanings as mentioned throughout the work unless specified explicitly.

Algorithm 2

```
Step 1: Randomly generate an initial swarm
Step 2: repeat
Step 3:         for each particle i do
Step 4:             if $f(x_i) > f(p_i)$ then $p_i \leftarrow x_i$
Step 5:             $p_g = \max(p_{neighbours})$
Step 6:             $v_{i+1} = w \times v_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (p_g - x_i)$
Step 7:             $x_i \leftarrow x_i + v_i$
Step 8:         end for
Step 9:     until termination criterion is met
```

Stochastic algorithms unlike other algorithms rely heavily on random number generators. Although true random number generation is not possible in any computers today, workable generators can always be achieved in computers. PSO algorithm also relies on random numbers. Random numbers vary the social, cognitive and inertial aspects of motion of particles in the PSO algorithm. This variation is actually the key to the success of PSO in any problem. The randomness leads to exploration of the search space and exploitation of the solution site. In the algorithm, the variables r_1 and r_2 are random numbers between 0 and 1. These have significance in the algorithm. These help produce refinedness in the algorithm.

Random number generators can be found in ANSI C but they do not help in producing numbers with many digits. The numbers generated keep on repeating after producing some numbers. Scientific researches require much more than that. Pure C library function for producing random numbers has this limitation because the range is limited. Python, on the other hand, is much preferred by researchers. The random number generator provided by Python library gives much more flexibility than C. To make the best from the facility provided by Python, this dissertation has the implementation in Python 2.7 in Linux (Ubuntu 10.04). The implemented code however runs in any system that runs Python.

4.2. OBJECT ORIENTED PARADIGM (OOP) IMPLEMENTATION

OOP is the contemporary technology in any program writing. The main features of OOP are inheritance, polymorphism and encapsulation and these have been thoroughly exploited in the implementation of PSO. A class called “particle” is defined with a constructor that initializes the class instance (object) with given attributes.

```
class Particle:
    position = []
    fitness = 0.0
    velocity = []
    bestPosition = []
    bestFitness = 0.0

    def __init__(self, position, fitness, velocity, \
        bestPosition, bestFitness):
        self.position = position
        self.fitness = fitness
        self.velocity = velocity
        self.bestPosition = bestPosition
        self.bestFitness = bestFitness
```

Figure 4.1 Python Code snippet for Class

Python class and the __init__ method.

A swarm of particles is produced by making array of “particle” objects. Many researches have varied the number of particles in a swarm and the number is not a fixed one to be settled at. Thus, arbitrary choice of 5 has been taken for this work. These objects are initialized with random values over the search space. The search space is defined by the objective function.

4.3. DATA STRUCTURES

The data structures used in the implementation are lists, tuples and arrays. Objects are used. Lists, arrays are readily supported by Python. Inbuilt methods supported by the data structures are heavily exploited.

4.4. INPUT AND OUTPUT

Other than the user choice of implementation, number of iterations, the number of particles, the choice of objective function and the choice of progression method of inertia weight, the input to the algorithm are the random numbers which are generated by the algorithm implicitly. Successive inputs are generated implicitly.

The outputs of the algorithm are numbers that are written in a file. Each number is the final optimal value on the user-supplied objective function. These numbers are later used for analysis purpose.

CHAPTER 5

5. OUTPUT

The program was run in a machine with following specifications:

Processor: Intel® Core™ i5 CPU M450 @2.40 GHz 2.40 GHz

Memory: 4.00 GB

System Type: 64 Bit

Operating System: Ubuntu 10.04

Manufacturer: Dell

Model: Studio 1569

Initial raw output was written on a file. Program was run many times with the parameters as listed in the first row of the following tables.

5.1. UNPROCESSED OUTPUT

All the outputs have not been listed because of their unprocessed nature and because each run of the program had 100 output lines, full output has not been provided here. However, interested readers can always contact the writer for details.

Function Parabola	Dimension = 5	Particles = 5	Iterations = 1000	Progression = Arithmetic
Inertia	Fitness	Point		
0.0:	16.8973834833	[-2.9620575489204048, 2.2715890459236325, 2.5383059164776665, -1.3523892468740488, -1.8918946097153899]		
0.01:	14.5692564608	[1.3595676184187615, 0.85760290133315198, -2.7213439725163147, 2.0709810563700009, -0.5391419690176702]		
Output truncated …				
0.98:	2.33162865947	[-0.486733176317931, -0.86020286155256442, -0.63722826640429719, -0.56851358861960299, -0.94264263059109954]		
0.99:	1.33301222257	[0.69304806424818777, -0.95276967836199278, 1.0924457885093641, 0.0, -0.41097460458049184]		

Table 5.1 Output: Parabola (5D AP)

Function = Parabola	Dimension = 5	Particles = 5	Iterations = 1000	Progression = Harmonic
Inertia	Fitness	Point		
1.0:	1.8895022298	[-0.46845357234277918, -0.23348256776660214, 0.67564548222344545, 1.1169622598130857, 0.42915734894218716]		
0.909090909091:	0.370785929137	[0.35955995369494342, 0.28830500074299614, 0.21224728311397278, 0.0, -0.51746876151426768]		
Output truncated …				
0.0925925925926:	3.68144527682	[-0.83670115368527165, 1.2816142994818707, -0.00072527219062334947, 2.0848292454224766, 0.045133281062593074]		
0.0917431192661:	4.56796456161	[-0.7601005663318473, -1.9610685532008929, -2.3704211971048665, 0.14561481813971344, -0.16715856156515257]		

Table 5.2 Output: Parabola(5D HP)