IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Parameters Selection, Applications & Convergence Analysis of PSO Algorithms

Sachin Kumar[1], Mr. N.K. Gupta[1]

Student, M.Tech-CSE, SSET, SHIATS, Allahabad, U.P, India[1]

Assistant Professor, CSE, SSET, SHIATS, Allahabad, U.P, India[2]

Abstract: A mathematical technique or methodology that deals with the finding of maximal or minimal of functions in some feasible searching space or region is called as Optimization. Every business or industry is involved in solving optimization problems. Some different varieties of optimization processes compete for the best minima solution. Particle Swarm Optimization (PSO) is a relatively new, advanced, and powerful method for optimization that has been empirically tested on many optimization problems and it perform well in solving those optimization problems. PSO is widely used to find the global optimum solution in a complex searching space. This work is focused on providing a review and discussion of the most established results on PSO algorithm as well as exposing the most active research topics that can encourage the practitioner for future work with improved results by applying little effort .This work introduces a theoretical concepts and some detailed explaining of the PSO algorithm, its advantages and disadvantages, judiciary selection of the various parameters with their effects. Moreover, this dissertation discusses a study of boundary conditions with the invisible wall technique, controlling the convergence behaviours of PSO, discrete-valued problems, multi-objective PSO, and applications of PSO. Finally, this work presents some kinds of improved versions as well as recent progress in the development of the PSO, and the future research issues are also discussed.

Keywords: Optimization, Swarm Intelligence, Social network, convergence, stagnation, multi-objective.

I. INTRODUCTION

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 behaviour of animals such as bird flocking, fish schooling and so on. It is based on the natural process of group communication to share individual knowledge when a group of birds or insects search food or migrate and so forth in a searching space, although all birds or insects do not know where the best position is. But from the nature of the social behaviour, if any member can find out a desirable path to go, the rest of the members will follow quickly.

The PSO algorithm basically learned from animal's activity or behaviour to solve optimization problems. 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 neighbours. 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.

The PSO method is becoming very popular because of its simplicity of implementation as well as ability to swiftly converge to a good solution. It does not require any gradient information of the function to be optimized and uses only primitive mathematical operators.

As compared with other optimization methods, it is faster, cheaper and more efficient. In addition, there are few parameters to adjust in PSO. That's why PSO is an ideal optimization problem solver in optimization problems. PSO is well suited to solve the non-linear, non-convex, continuous, discrete, integer variable type problems.

Swarm intelligence (SI) is based on the collective behaviour of decentralized, self-organized systems. It may be natural or artificial. Natural existing examples of SI are ant colonies system, schooling of fishes, bird flocking, bee swarming and so on. Besides multi-robot systems, some computer program for tackling optimization and data analysis problems are examples for some human artifacts of SI. The most successful SI techniques are Particle Swarm Optimization PSO and Ant Colony Optimization (ACO). In PSO, each particle flies through the multidimensional space and adjusts & updates its position in every step with its own individual experience and that of peers toward an optimum solution by the entire swarm. Thus the PSO is a member of the Swarm Intelligence.

1.3 Objectives

This dissertation aims to answer the following questions:

Q.1: How can premature convergence and stagnation problems in the PSO algorithm be prevented?

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Q.2: When and how particles are again able to reinitialize?
Q.3: For the PSO algorithm, what will be the consequence if

a) The maximum velocity V_{max} is too large or small?
b) The acceleration coefficients c_1 and c_2 are equal or not?
c) The acceleration coefficients c_1 and c_2 are very large or small?

Q.4: How can the boundary problem in the PSO method be solved?
Q.5: How can the discrete-valued problems are solved by the PSO method?
This paper is based on answering these questions in a briefs descriptions.

Optimization: Optimization determines the best-suited solution to problem under given circumstances. For example, a manager needs to take many technological and managerial plans at several times. The final goal of the plans is either to minimize the effort required or to maximize the desired benefit. Optimization refers 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 interchangeably. For this reason, now-a-days, it is very important in many professions.
Optimization problems may be linear (called linear optimization problems) or non-linear (called non-linear optimization problems). Non-linear optimization problems are generally very difficult to solve.
Based on the problem characteristics, optimization problems are classified in the following:

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 types of problems are said to be constrained optimization problems and defined as

$$\begin{aligned} &\textbf{minimize} && \mathbf{f(x), x = (x_1, x_2, x_3, \dots, x_n)} \\ &\textbf{subject to} && \mathbf{g_m(x) \le 0, m = 1, 2, \dots\dots, n_g} \\ & && \mathbf{h_m(x) = 0, m = n_g{+}1, \dots, n_g{+}n_h} \\ & && \mathbf{\forall\, x \in R^n} \qquad \textbf{(1)} \end{aligned}$$

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 and defined as

$$\textbf{minimize } \mathbf{f(x), x}\ \forall\ \mathbf{R} \qquad \textbf{(2)}$$

where n is the dimension of x.

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 and defined as

$$\begin{aligned} &\textbf{minimize} && \mathbf{f(x, \varpi(t)), x = (x1, x2, x3, \dots, xn), \varpi(t)} \\ & && \mathbf{= (\varpi_1(t), \varpi_2(t), \dots, \varpi_{n\varpi}(t))} \\ &\textbf{subject to} && \mathbf{g_m(x) \le 0, m = 1, 2, \dots\dots, n_g} \\ & && \mathbf{h_m(x) = 0, m = n_g{+}1, \dots, n_g{+}n_h} \\ & && \mathbf{\forall\, x \in R^n} \qquad \textbf{(3)} \end{aligned}$$

where ϖ(t) is a vector of time-dependent objective function control parameters, x*(t) is the optimum found at time step t. There are two techniques to solve the problems: Global & Local optimization techniques.

Global Optimization
A global minimizer is defined as x* such that

$$\mathbf{f(x^*) \le f(x), \forall x \in S} \qquad \textbf{(4)}$$

Where S is the search space & S = R^n for unconstrained problems.
Here the term global minimizer refers to the value $f(x*), x*$ is called the global minimizer.Some global minimizer requires a starting point $z_0 \in S$ and it will be able to find the global minimizer $x*$ if $z_0 \in S$.

Local Optimization: A local minimizer $x*_L$ of the region L, is defined as

$$\mathbf{f(x^*) \le f(x), \forall x \in L} \qquad \textbf{(5)}$$

Where L is the subset of R^n.
Here, a local optimization method should guarantee that a local minimizer of the set is found.
Finally, local optimization techniques try to find a local minimum and its corresponding local minimizer, whereas global optimization techniques seek to find a global minimum or lowest function value and its corresponding global minimizer.

The Basic Model of PSO algorithm
Kennedy and Eberhart first established a solution to the complex non-linear optimization problem by imitating the behaviour of bird flocks. They generated the concept of function-optimization by means of a particle swarm. Consider the global optimum of an n-dimensional function defined by

$$\mathbf{f(x) = (x1, x2, x3, \dots, xn) = f(X)} \qquad \textbf{(6)}$$

Where x_i is the search variable, which represents the set of the free of the given functions. The aim is to find a value $x*$ such that the function $f(x*)$ is either a maxima or a minima in the search space.
Consider a function given by

$$f_1 = (x_1 + x_2)^2 \qquad (7)$$

And $$f_2 = x_1 \sin(4*\pi*x_1) - x_2 \sin(4*\pi*x_2) \qquad (8)$$

DOI 10.17148/IJARCCE.2016.5815

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

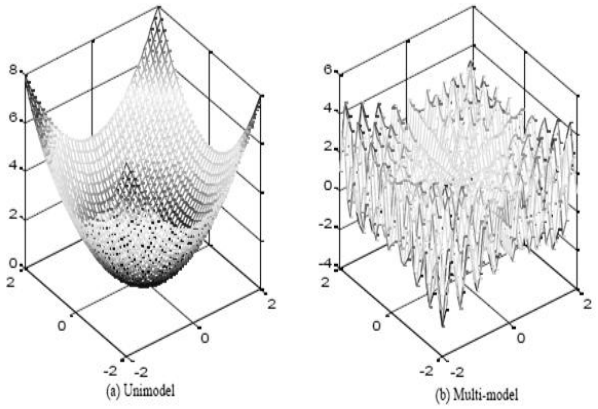

Figure 3.1: Plot of the functions f_1 and f_2.

From the figure 1(a), it is the clear that the global minima of the function f_1 is at $(x_1,x_2) = (0,0)$,i.e., at the origin of function f_1 in the search space. That means it is a unimodel function, which has only one minimum. However, to find the global optimum is not so easy for multi-model functions, which have multiple local minima. Figure 1 (b) shows the function which has a rough search space with multiple peaks, so many agents have to start from different initial locations and continue exploring the search space until at least one agent reach the global optimal position. During this process all agents can communicate and share their information among themselves. This thesis discusses how to solve the multi-model function problems.

The Particle Swarm Optimization (PSO) algorithm is a multi-agent parallel search technique which maintains a swarm of particles and each particle represents a potential solution in the swarm. All particles fly through a multidimensional search space where each particle is adjusting its position according to its own experience and that of neighbours. Suppose denote the position vector of particle in the multidimensional search space (i.e.) at time step , then the position of each particle is updated in the search space by

$$\mathbf{x_i^{t+1} = x_i^t + v_i^{t+1} \text{ with } x_i^0}$$

where,v_i^t is the velocity vector of particle that drives the optimization process and reflects both the own experience knowledge and the social experience knowledge from the all particles; $U(x_{max}, x_{min})$ is the uniform distribution where x_{max} and x_{min} are its minimum and maximum values respectively.

Therefore, in a PSO method, all particles are initiated randomly and evaluated to compute fitness together with finding the personal best (best value of each particle) and global best (best value of particle in the entire swarm). After that a loop starts to find an optimum solution. In the loop, first the particles' velocity is updated by the personal and global bests, and then each particle's position is updated by the current velocity. The loop is ended with a stopping criterion predetermined in advance.
Basically, two PSO algorithms, namely the Global Best (gbest) and Local Best (lbest) PSO, have been developed which differ in the size of their neighbourhoods. These algorithms are discussed in Sections respectively.

3.1.1 Global Best PSO

The global best PSO (or gbest PSO) is a method where the position of each particle is influenced by the best-fit particle in the entire swarm. It uses a star social network topology where the social information obtained from all particles in the entire swarm. In this method each individual particle,i∈[1,,,,,n] where n>1 has a current position in search space x_i,a current velocity,v_i,and a personal best position in search space,$P_{best,i}$ corresponds to the position in search space where particle i has the smallest value as determined by the objective function f , considering a minimization problem. In addition, the position yielding the lowest value amongst all the personal best is called the global best position which is denoted by G_{best} . The following equations define how the personal and global best values are updated, respectively.
Considering minimization problems, then the personal best position $P_{best,i}$ at the next step,t+1,where t ∈ [0,,,,N],is calculated as

$$\mathbf{P_{best,i}^{t+1}} = \begin{cases} \mathbf{P_{best,i}^t} & \text{if } f\left(\mathbf{x_i^{t+1}}\right) > \mathbf{P_{best,i}^t} \\ \mathbf{x_i^{t+1}} & \text{if } f(\mathbf{x_i^{t+1}}) \le \mathbf{P_{best,i}^t} \end{cases} \quad \mathbf{(9)}$$

Where $R^n \rightarrow R$ is the fitness function.
The global best position G_{best} at time step t, is calculated as

$$\mathbf{G_{best} = min\ \{P_i^{best}\}, where\ i\ \in [1, \dots, n]\ and\ n > 1} \quad \mathbf{(10)}$$

Therefore it is important to note that the personal best $P_{best,i}$ is the best position that the individual particle i has visited since the first time step. On the other hand, the global best position G_{best} is the best discovered position by any of the particle in the entire swarm.

For gbest PSO method, the velocity of particle i is calculated by

$$\mathbf{v_{ij}^{t+1} = v_{ij}^t + c_1 r_{1j}^t \left[P_{best,i}^t - x_{ij}^t\right] + c_2 r_{2j}^t \left[G_{best} - x_{ij}^t\right]} \quad \mathbf{(11)}$$

v = is the velocity vector of particle i in dimension j at time t
x_{ij}^t = is the position vector of particle i in dimension j at time t
$P_{best,i}$ = is the personal best position of particle i in dimension j found from initialization through time t
G_{best} = is the global best position of particle i in dimension j found from initialization through time t
c_1 & c_2 = are positive acceleration constants which are used to level the contributions of the cognitive and social components respectively
r_{1j}^t & r_{2j}^t = are random numbers from uniform distributions U(0,1) at time t.

DOI 10.17148/IJARCCE.2016.5815

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Local Best PSO

The local best PSO (or lbest PSO) method only allows each particle to be influenced by the best-fit particle chosen from its neighbourhood, and it reflects a ring social topology. Here this social information exchanged within the neighbourhood of the particle, denoting local knowledge of the environment. In this case, the velocity of particle is calculated by

$$\mathbf{v_{ij}^{t+1}} = \mathbf{v_{ij}^{t}} + \mathbf{c_1 r_{1j}^{t}}\left[\mathbf{P_{best,i}^{t}} - \mathbf{x_{ij}^{t}}\right] + \mathbf{c_2 r_{2j}^{t}}\left[\mathbf{L_{best}} - \mathbf{x_{ij}^{t}}\right] \quad (12)$$

Where $L_{best,i}$ is the best position that any particle has had in the neighbourhood of particle i found from initialization through time t .

Parameters:

Swarm size

Swarm size or population size is the number of particles n in the swarm. A big swarm generates larger parts of the search space to be covered per iteration. A large number of particles may reduce the number of iterations need to obtain a good optimization result. In contrast, huge amounts of particles increase the computational complexity per iteration, and more time consuming. From a number of empirical studies, it has been shown that most of the PSO implementations use an interval of for the swarm size.

Iteration numbers

The number of iterations to obtain a good result is also problem-dependent. A too low number of iterations may stop the search process prematurely, while too large iterations has the consequence of unnecessary added computational complexity and more time needed.

Velocity Components

The velocity components are very important for updating particle's velocity. There are three terms of the particle's velocity in equations mentioned above:

1. The term $\mathbf{v_{ij}^{t}}$ is called inertia component that provides a memory of the previous flight direction that means movement in the immediate past. This component represents as a momentum which prevents to drastically change the direction of the particles and to bias towards the current direction.
2. The term $\mathbf{c_1 r_{1j}^{t}}\left[\mathbf{P_{best,i}^{t}} - \mathbf{x_{ij}^{t}}\right]$ is called cognitive component which measures the performance of the particles i relative to past performances. This component looks like an individual memory of the position that was the best for the particle. The effect of the cognitive component represents the tendency of individuals to return to positions that satisfied them most in the past. The cognitive component referred to as the nostalgia of the particle.
3. The term $\mathbf{c_2 r_{2j}^{t}}\left[\mathbf{G_{best}} - \mathbf{x_{ij}^{t}}\right]$ for **gbest** PSO or $\mathbf{c_2 r_{2j}^{t}}\left[\mathbf{G_{best}} - \mathbf{x_{ij}^{t}}\right]$ for **lbest** PSO is called social component which measures the performance of the particles i relative to a group of particles or neighbours. The social component's effect is that each particle flies towards the best position found by the particle's neighbourhood.

Acceleration coefficients

The acceleration coefficients c_1 & c_2 and , together with the random values r_1 and r_2 , maintain the stochastic influence of the cognitive and social components of the particle's velocity respectively. The constant c_1 expresses how much confidence a particle has in itself, while c_2 expresses how much confidence a particle has in its neighbours. There are some properties of c_1 & c_2.

- When $c_1 = c_2 = 0$ then all particles continue flying at their current speed until they hit the search space's boundary. Therefore ,from the equation the velocity update equation is calculated as

$$\mathbf{v_{ij}^{t+1}} = \mathbf{v_{ij}^{t}} \quad (13)$$

- When $c_1 > 0$ & $c_2 = 0$,all particles are independent. The velocity update will be

$$\mathbf{v_{ij}^{t+1}} = \mathbf{v_{ij}^{t}} + \mathbf{c_1 r_{ij}^{t}}\left[\mathbf{P_{best,i}^{t}} - \mathbf{x_{ij}^{t}}\right] \quad (14)$$

On the contrary, when $c_2 > 0$ & $c_1 = 0$,all particles are attracted to a single point (i.e., G_{best}) in the entire swarm and the update velocity will become

$$\mathbf{v_{ij}^{t+1}} = \mathbf{v_{ij}^{t}} + \mathbf{c_1 r_{ij}^{t}}\left[\mathbf{G_{best,i}^{t}} - \mathbf{x_{ij}^{t}}\right] \quad (15)$$

$$\mathbf{v_{ij}^{t+1}} = \mathbf{v_{ij}^{t}} + \mathbf{c_1 r_{ij}^{t}}\left[\mathbf{L_{best,i}^{t}} - \mathbf{x_{ij}^{t}}\right] \quad (16)$$

- When $\mathbf{c_1 = c_2}$,all particles are attracted towards the average of $\mathbf{P_{best,i}^{t}}$ **and** $\mathbf{G_{best,i}^{t}}$.
- When $c_1 >> c_2$, each particle is more strongly influenced by its personal best position, resulting in excessive wandering. In contrast, when then all particles are much more influenced by the global best position, which causes all particles to run prematurely to the optima.

Improvements Factors & Analysis of Convergence:

Usually, the particle velocities build up too fast and the maximum of the objective function is passed over. In PSO, particle velocity is very important, since it is the step size of the swarm. At each step, all particles proceed by adjusting the velocity that each particle moves in every dimension of the search space. There are two characteristics: exploration and exploitation. Exploration is the ability to explore different area of the search space for locating a good optimum, while exploitation is the ability to concentrate the search around a searching area for refining a hopeful solution. Therefore these two characteristics have to balance in a good optimization algorithm. When the velocity increases to large values, then particle's positions update quickly. As a result, particles leave the boundaries of the search space and

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

diverge. Therefore, to control this divergence, particles' velocities are reduced in order to stay within boundary constraints. The following techniques have been developed to improve speed of convergence, to balance the exploration-exploitation trade-off, and to find a quality of solutions for the PSO:

Velocity clamping

Eberhart and Kennedy first introduced velocity clamping; it helps particles to stay within the boundary and to take reasonably step size in order to comb through the search space. Without this velocity clamping in the searching space the process will be prone to explode and particles' positions change rapidly. Maximum velocity controls the granularity of the search space by clamping velocities and creates a better balance between global exploration and local exploitation

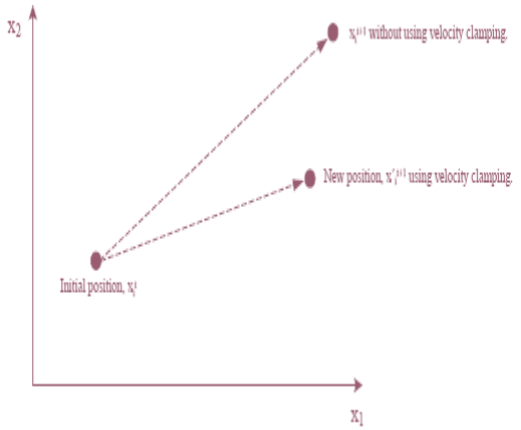

Figure 2: Illustration of effects of Velocity Clamping for a particle in a two-dimensional search space

Figure 4.1 illustrates how velocity clamping changes the step size as well as the direction when a particle moves in the process. In this figure , $\mathbf{x_i^{t+1}}$**and** $\mathbf{x'^{t+1}_i}$denote respectively the position of particle i without using velocity clamping and the result of velocity clamping.

Now if a particle's velocity goes beyond its specified maximum velocity V_{max} this velocity is set to the value V_{max} and then adjusted before the position update by,

$$\mathbf{v_i^{t+1} = min\ (v'^{t+1}_i , V_{max})} \qquad \mathbf{(17)}$$

where, v'^{t+1}_i is calculated using equations given above

If the maximum velocity V_{max} is too large , the particle may move erratically and jump over the optimum solution .On the other hand if the V_{max} is too small , the particle's moment is limited and the swarm may n ot explore efficiently or the swarm may become trapped in the local optimum. This problem can be solved when the maximum velocity is calculated by a fraction of the domain of the search space on each dimension by subtracting the lower bound from the upper bound, and is defined as

$$\mathbf{V_{max} = \varepsilon\ (x_{max} - x_{min})} \qquad \mathbf{(18)}$$

Where $\mathbf{x_{max}}$ & $\mathbf{x_{min}}$ are respectively the maximum and minimum values of x, and $\varepsilon \in [0,1]$. For example, if ε = 0.5 and x = [-150,150] on each dimension of the search space. Then the range of the search space is 300 per dimension and velocities, are then clamped to a percentage of that range according to equation, then the maximum velocity is V_{max} = 150.

There is another problems when all velocities becomes equal to the maximum velocity$\mathbf{V_{max}}$. To solve this problem, V_{max} can be reduced over time. The initial steps starts with the large values of $\mathbf{V_{max}}$, and then it is decreased it over time. The advantage of velocity clamping is that it controls the explosion of the velocity in the searching space. On the other hand, the disadvantage is that, the best value of V_{max} should be chosen for each different optimization problem using empirical techniques and finding the accurate value for the V_{max} for the problem being solved is very critical and not simple, as a poorly chosen $\mathbf{V_{max}}$ can lead to extremely poor performance.

Finally, $\mathbf{V_{max}}$ was introduced to prevent explosion and divergence. However, it has become unnecessary for convergence because of the use of the inertia weight ω and constriction factor χ.

4.1.2. Inertia-weight:

The inertia-weight, denoted by ω, is considered to replace V_{max} by adjusting the influence of of the previous velocities in the process, i.e., it controls the momentum of the particles by weighing the contribution of the previous velocity. The inertia-weight 'ω' will at every step be multiplied by the velocity at the previous time step, i.e., $\mathbf{v^t_{ij}}$. Therefore, in the gbest PSO, the velocity equation of the particle i with the inertia-weight changes from equation to

$$\mathbf{v_{ij}^{t+1} = \omega v_{ij}^t c_1 r_{1j}^t \left[P_{best,i}^t - x_{ij}^t\right] + c_2 r_{2j}^t \left[G_{best} - x_{ij}^t\right]} \mathbf{(19)}$$

In the lbest PSO, the velocity equation changes in a similar way as the above velocity equation do.

The inertia-weight was first introduced by Shi & Eberhart in 1999 to reduce the velocity over time, to control the exploitations and explorations ability of the swarm, and to converge the swarm more accurately and efficiently compared to the equation. If ω is greater or equal to 1, then the velocity increases over time and particle can hardly change their direction to move back towards optimum, and the swarm diverges. If ω << 1, then little moment is only saved from the previous step and quick changes of directions are to set in the process. If ω = 0, particle velocity vanishes and all particles move without knowledge of the previous velocity in each step.

The inertia weight can be implemented as a fixed value or dynamically change values. Initial implementations of ω used as a fixed value for the whole process for the all particles. But now dynamically inertia value is used because this parameters controls the exploration and

DOI 10.17148/IJARCCE.2016.5815

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

exploitation of the search space. Usually the large inertia value is high at first, which allows all the particles to move freely in the search space at the initial step and decreases over time. Therefore the process is shifted from the explorative mode to the exploitative mode. This decreasing inertia weight has produced good results in many optimization problems. To control the balance between local and global exploration, to obtain faster convergence, and to reach an optimum, the inertia weight whose value decreases linearly with the iteration number is set according to the following equation

$$\omega^{t+1} = \omega_{max} - \left(\frac{\omega_{max} - \omega_{min}}{t_{max}}\right) t \ , \ \omega_{max} > \omega_{min} \quad (20)$$

where ω_{max} & ω_{min} are the initial and the final values of the inertia weight respectively.
t_{max} is the maximum iteration number.
And t is the current iteration number.

Generally, the inertia weight ω decreases from 0.9 to 0.4 over the entire process.
Van der Bergh and EngelBrechet, Trelea have defined a condition, that is

$$\omega > \frac{1}{2}(c_1 + c_2) - 1 \quad (21)$$

guarantees the convergence. Divergent or cyclic behaviour can occur in this process if this condition is not satisfied. The inertia-weight technique is very useful to ensure convergence. However, there is a disadvantage of this method is that once the inertia weight is decreased, it cannot increase if the swarm needs to search the new areas. This method is not able to recover its exploration mode.

4.1.3. Constriction Factor:

This technique introduced a new parameter 'χ' known as the constriction factor. The constriction factor coefficient was developed by Clerc. This coefficient is extremely important to control the exploration and exploitation tradeoffs, to ensure the convergence behaviour, and also to exclude the inertia weight ω and the maximum velocity V_{max}. Clerc's proposed velocity updation of the particle i for the j dimension is calculated as,

$$v_{ij}^{t+1} = \chi[v_{ij}^{t} + \emptyset_1(P_{best,i}^{t} - x_{ij}^{t}) + \emptyset_2(G_{best} - x_{ij}^{t})] \quad (22)$$

where

$$\chi = \frac{2}{\left|2 - \emptyset - \sqrt{\emptyset^2 - 4\emptyset}\right|},$$

$\emptyset = \emptyset_1 + \emptyset_2$

$\emptyset_1 = c_1 r_1$

And $\emptyset_2 = c_2 r_2$.

If ϕ < 4, then all the particles would slowly spiral towards around the best solution in the searching space without the convergence guarantee. If ϕ > 4, then all particles converged faster and guaranteed.

Eberhart & Shi empirically illustrated that if the velocity clamping and constriction coefficient are used together, that tends to obtain the faster convergence rate.

Guaranteed Convergence PSO:

When the current position of the particle coincides with the global best position, then the moves away from this point if its previous velocity is non-zero. In other words, when $x^t_{ij} = P^t_{best,i} = G^t_{best,i}$, then the velocity updates depends only on the value of ωv^t_{ij}. Now if the previous velocities of particles are close to zero, all particles stop moving once and they catch up with the global best position, which can lead to the premature convergence of the processes. This doesn't even guaranteed that the process has converged to a local minima, it only means that all the particles have converged to the best position in the entire swarm. This lead to the stagnation of the search process which the PSO algorithm can overcome by forcing the global best position to change when $x^t_{ij} = P^t_{best,i} = G^t_{best,i}$.

4.2. Boundary Condition:

Sometimes, it is important that the search space has some limitations for swarm to avoid exploding. In other words, particles are allowed to fly randomly beyond the search space and generate the invalid solutions but only sometimes. Generally, velocity clamping technique is used to limit the velocity of the particle to the maximum velocity V_{max} . The maximum velocity, V_{max} the inertia weight ω, and the constriction coefficients value χ do not always confines the particles to the solution space. These parameters are not able to provide the information of the space in which the searching particles stays. Besides, some particles still runs away from the solution space even having good choice of parameters V_{max}.

There are two main difficulties connected with previous velocity techniques:

1) The choice of the suitable value for V_{max} can be non-trivial and also very important for the overall performance of the method.
2) The previous velocity techniques cannot provide information about how the particles are enforced to stay within the selected space all the time.
To overcome this problem, boundary conditions of the PSO algorithm, this will be parameter free, efficient and also reliable.
To solve this problem, different types of the boundary conditions (BC) have been introduced and the unique features that distinguish each boundary condition are showed in fig .

These boundary conditions forms two groups:

1. Restricted BC: This includes namely, absorbing, reflecting, and damping.
2. Unrestricted BC: This includes namely, invisible, invisible/reflecting, and invisible/damping.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

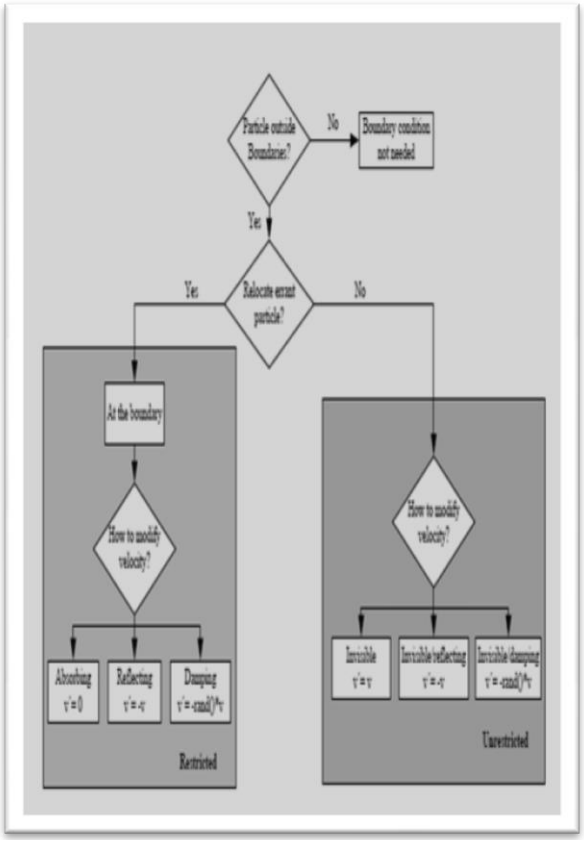

Figure 3: Various boundary conditions in PSO

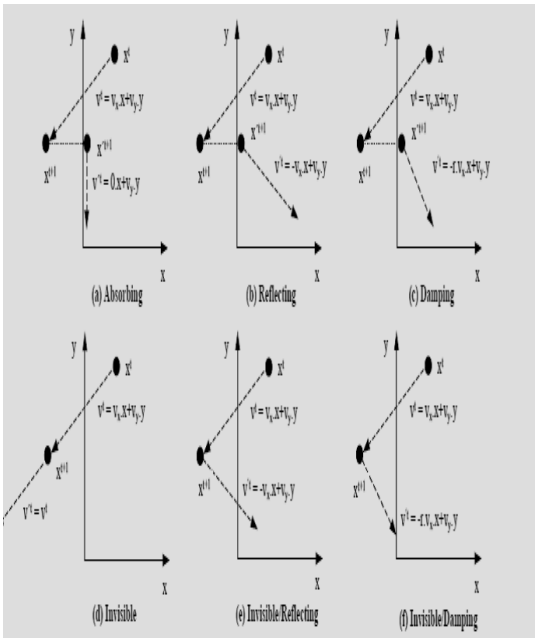

Fig 4: Six different BCs for a 2D search space, 'x' & 'v' represent the modified position and the velocity respectively and r is a random factor [0,1].

The six BCs are discussed as follows:

1. Absorbing Boundary Condition:
When a particle goes outside the solution space in one of the dimensions, the particle is relocated at the wall of the solution space and the velocity of the particle is set to zero in that dimension as illustrated in fig. 4(a). This means that, in this condition, such kinetic energy of the particle is absorbed by the soft wall so that the particle will return to the solution space to find the optimum solution.

2. Reflecting Boundary Condition:
When a particle goes outside the solution space in one of the dimensions, the particle is relocated at the wall of the solution space and the sign of the velocity of the particle is changed in the opposite direction in that dimension as illustrated in fig. 4(b), This means that the particle is reflected by the hard wall and then it will move back towards he solution space to find the optimum solution.

3. Damping Boundary Condition:
When a particle goes outside the solution space in one of the dimensions, the particle is relocated at the wall of the solution space and the sign of the velocity of the particle is changed in the opposite direction in that dimension with a random coefficient between 0 and 1 as illustrated in fig 4(c). Thus Damping BC acts very similar as the reflecting boundary condition except randomly determined part of the energy is lost because of the imperfect reflection.

4. Invisible Boundary Condition:
In this condition, the particle is considered to stay outside the solution space, while the fitness evaluation of that position is skipped and a bad fitness value is assigned to it as illustrated in fig 4(d). Thus the attraction of the personal and global best positions will counteract the particle's momentum, and ultimately pull it back inside the solution space.

5. Invisible/Reflecting Boundary Condition:
In this condition, the particle is considered to stay outside the solution space, while the fitness evaluation of that position is skipped and a bad fitness value is assigned to it as illustrated in fig 4(e). Also, the sign of the velocity of the particle is changed in the opposite direction so that the momentum of the particle is reversed to accelerate it back toward in the solution space.

6. Invisible/Damping Boundary Condition:
In this condition, the particle is considered to stay outside the solution space, while the fitness evaluation of that position is skipped and a bad fitness value is assigned to it as illustrated in fig 4(f). Also, the velocity of the particle is changed in the opposite direction with a random coefficient between 0 and 1 in that dimension so that the reversed momentum of the particle which accelerates it back towards in the solution space is damped.

Results:
To solve this problem, anew parameter is introduced to the PSO algorithm. Let T be the index of the global best particle, so that

$$y_\tau = G_{best} \qquad (23)$$

DOI 10.17148/IJARCCE.2016.5815

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

A new velocity update equation for the globally best positioned particle, y_T has been suggested to keep y_T moving until it has reached the local minimum

$$\mathbf{v_{\tau j}^{t+1} = -x_{\tau j}^{t} + G_{best}^{t} + \omega v_{\tau j}^{t} + \rho^{t}(1 - 2r_{2j}^{t})} \quad \textbf{(24)}$$

where

$\boldsymbol{\rho^t}$ Is a scaling factor and causes the PSO to perform a random search in an area surrounding The global best position P_{best} .

$\mathbf{-x_{\tau j}^{t}}$ Reset the particle position to the position $\mathbf{G_{best,i}^{t}}$

$\boldsymbol{\omega v_{\tau j}^{t}}$ Represents the current search direction.

$\boldsymbol{\rho^t(1 - 2r_{2j}^{t})}$ Generates a random sample from a sample space with side length $\mathbf{2\rho^t}$

Combining the position update equation (3.4) and the new velocity update equation for the global best particle τ yields the new position update equation

$$\mathbf{x_{\tau j}^{t+1} = G_{best}^{t} + \omega v_{\tau j}^{t} + \rho^{t}(1 - 2r_{2}^{t})} \quad \textbf{(25)}$$

While all the other particle in the swarm continue using the normal current velocity update and the position update equation respectively.

The parameter rho controls the diameters of the search space and the value of rho is adapted after each time step using

$$\rho^0 = 1.0$$

$$\rho^{t+1} = \begin{cases} 2\rho^t & \begin{cases} \textbf{if } \#\textbf{successes}\,(t) > \epsilon_s \\ \textbf{if } \#\textbf{failures}\,(t) > \epsilon_f \end{cases} \\ 0.5\rho^t & \\ \rho^t & \textbf{otherwise} \end{cases} \quad \textbf{(26)}$$

Where #successes and #failures respectively denote the number of consecutive successes and failures, and a failures is defined as $f(G_{best}^{t+1}) = f(G_{best}^{t})$. The following conditions must also be implemented to ensure that the equation is well defined:

#successes (t+1) > #successes(t) → #failures(t+1) = 0

And

#failures (t+1) > #failures(t) → #successes(t+1) = 0 (27)

Therefore, when a success occurs, the failures count set to zero and similarly when a failure occurs, then the success count is reset.

GCPSO uses adaptive ρ to obtain the optimal of the sampling volume given the current state of the algorithm. If a specific value of ρ repeatedly results in a success, then a large sampling volume is selected to increase the maximum distance travelled in one step. On the other hand, when ρ produces ϵ_f consecutive failures, then the sampling volume is too large and must be consequently reduced. Finally, stagnation can be totally prevented if $\rho^t > 0$ for all steps.

Application of Particle Swarm Optimization:

This section discusses the various application areas of the PSO methodology.

Kennedy & Eberhart first established the practical application PSO in 1995 in the field of neural network training and was reported together with the algorithm itself. PSO have been successfully used across a wide range of applications, for instance, telecommunication, system control, design, data mining, etc. The modern developed PSO is used for the advanced problematic scenarios whereas the original PSO algorithm was used mainly to solve the unconstrained, single-objective optimization problems.

Various areas where PSO is applied are:

6.1. ANTENNAS DESIGN:

The optimal control and designed of the phased arrays, broadband antenna design and modeling, reflector antenna, optimization of the reflect array antenna, antenna modeling, design of a periodic antenna arrays, near-field antenna measurement, synthesis of antenna arrays, adaptive antenna arrays, design of implantable antennas and so on.

6.2. SIGNAL PROCESSING:

Pattern recognition of the flatness signal, design of IIR filters, 2D IIR filters, speech coding, non-linear adaptive filter, Costas arrays, blind detection, bind source separation, distributed odour source localization, and so on.

6.3. NETWORKING:

Radar network, Bluetooth network, TCP network control, WDM telecommunication control, grouped and delayed broadcasting, bandwidth reservation, transmission network planning, voltage regulation, network reconfiguration and expansion, economic dispatch problem, distributed generation.

6.4. IMAGE & GRPAHICS:

Planning landmarks in orthodontic x-ray images, pedestrian detection & tracking, stop-sign detection, image registration, microwave imaging, pixel classification, texture synthesis, scene matching, contrast enhancement, character recognition, image noise cancellation, and so on.

6.5. POWER GENERATION & CONTROLLING:

Automatic generation control, power transformer protection, power loss minimization, load forecasting, STATCOM power system, hybrid power generation system, power system performance optimization, secondary voltage control, power control & optimization, large-scale power plant control, control of photovoltaic system, analysis of power control signals, generation planning and restructuring, production costing, and so on.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

6.6. FUZZY SYSTEMS, CLUSTERING & DATA-MINING:

Design of neurofuzzy networks, fuzzy rule extraction, fuzzy control, fuzzy modeling, design of hierarchical fuzzy systems, dimensionality reduction, documents and information clustering, dynamic clustering, cascading classifiers, fuzzy clustering, data mining, feature selection, and so on.

6.7. OPTIMIZATION:

Electrical motor optimization, optimization of internal combustion engines, floor planning, packing & knapsack, knight cover problems, layout optimization, path optimization, urban planning, FPGA placement and routing.

6.8. PREDICTION & FORECASTING:

Water quality prediction & classification, predicition of chaotic system, streamflow forecast, ecological models, electric load forecasting, time series predicition, battery pack state of charge estimation, predicition of elephant migration, urban traffic flow forecasting, and so on.

6.9. ROBOTICS:

Control of robotics manipulators and arms, motion planning and control, odour source localization, swarm robotics, unmanned vehicle navigation, path planning, unsupervised robotic learning, environment mapping, voice control of robots, collective robotic search, obstacle avoidance, soccer playing, transport robot, and so on.

6.10. DESIGN & MODELLING:

Conceptual design, electromagnetic case, induction heating cooker design, VLSI design, power system, RF circuit synthesis, worst case electronic design, motor design, antenna design, transmission lines, identifying ARMAX models, power plant and systems, chaotic time series modelling, model order reduction, ultra wideband channel modelling, fliter design, thermal process system identification, and so on.

6.11. BIOMEDICAL:

Human tremor analysis for the diagnosis of Perkinson's disease, inference of gene regulatory networks, RNA secondary structure determination, biomechanics optimization, DNA motif detection, cancer classification, survival prediction, biomarker selection, protein structure predicition and docking, drug design, radio therapy planning, analysis of brain magneto encephalography data, phylogenetic tree reconstruction, and so on.

CONCLUSION

This paper discussed the basic Particle Swarm Optimization algorithm, geometrical and mathematical explanation of PSO, particles' movement and the velocity update in the search space, the acceleration coefficients and particles'. A set of convergence techniques, i.e. velocity clamping, inertia weight and constriction coefficient techniques which can be used to improve speed of convergences and control the exploration and exploitation abilities of the entire swarm, was illustrated. The Guaranteed Convergence PSO (GCPSO) algorithm was analyzed. This algorithm is very important to solve a problem when all particles face premature convergence or stagnation in the search process. Boundary conditions were presented which are very useful in the PSO algorithm.

The Multi-Start PSO (MSPSO) algorithm attempts to detect when the PSO has found lack of diversity. Once lack of diversity is found, the algorithm re-starts the algorithm with new randomly chosen initial positions for the particles. The Multi-phase PSO (MPPSO) algorithm partitions the main swarm into sub-swarms or subgroups, where each sub-swarm performs a different task, exhibits a different behaviour and so on. Then the swarms cooperate to solve the problem by sharing the best solutions they have discovered in their respective sub-swarms. During the optimization process, high speed of convergence sometimes generates a quick loss of diversity which lead to undesirable premature convergence. To solve this problem, the perturbed particle swarm algorithm (PPSO) illustrated in this chapter. The Multi-Objective PSO (MOPSO) algorithm is very important when an optimization problem has several objective functions. One discrete optimization problem was solved by the Binary PSO (BPSO) algorithm.

The PSO algorithm has some problems that ought to be resolved. Therefore, the future works on the PSO algorithm will probably concentrate on the following:

1. Find a particular PSO algorithm which can be expected to provide good performance.
2. Combine the PSO algorithm with other optimization methods to improve the accuracy.
3. Use this algorithm to solve the non- convex optimization problems.

REFERENCES

[1] Ioan Cristian Trelea, "The particle swarm optimization algorithm: convergence analysis & parameters selection ," in Elsevier-Information Processsing Letters 2002, PII: S0020-0190(02)00447-7.

[2] Praveen Kumar Tripathi, Sanghamitra Bandyopadhyay, Sankar Kumar Pal, "Multi-Objective Particle Swarm Optimization with time variant inertia and acceleration coefficients ," in Elsevier-doi:10.1016/j.ins.2007.06.018.

[3] Maurice Clerc and James Kennedy, "The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space," in IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 6, NO. 1, FEBRUARY 2002.

[4] D. Bratton and J. Kennedy, "Defining a Standard for Particle Swarm Optimization ," in IEEE Swarm Intelligence Symposium, 2007, pp. 120-127.

[5] X. Li and K. Deb, "Comparing lbest PSO Niching algorithms Using Different Position Update Rules," in In Proceedings of the IEEE World Congress on Computational Intelligence, Spain, 2010, pp. 1564-1571.

[6] M. Dorigo and M. Birattar, "Swarm intelligence," in Scholarpedia, 2007, pp. 2(9):1462.

[7] Andries P. Engelbrecht, Computational Intelligence: An Introduction: John Wiley and Sons, 2007, ch. 16, pp. 289-358.
[8] Riccardo Poli, "Review Article-Analysis of the Publications on the Applications of Particle Swarm Optimisation," Journal of Artificial Evolution and Applications, p. 10, 2008.
[9] Singiresu S. Rao, Engineering Optimization Theory and Practice, 4th edition, Ed.: John Wiley and Sons, 2009.
[10] El-Ghazali Talbi, Metaheuristics-From Design to Implementation: John Wiley and Sons, 2009.
[11] George I. Evers, An Automatic Regrouping Mechanism to Deal with Stagnation in Particle Swarm Optimization(Master's thesis), The University of Texas-Pan American., 2009, Department of Electrical Engineering.
[12] Anthony Carlisle and Gerry Dozier, "An Off-The-Shelf PSO," in Workshop Particle Swarm Optimization, Indianapolis, 2001.
[13] Nanbo Jin and Yahya Rahmat-Samii, "Advances in Particle Swarm Optimization for Antenna Designs: Real-Number, Binary, Single-Objective and Multiobjective Implementations," IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, vol. 55, no. 3, pp. 556-567, MARCH 2007.
[14] F. van den bergh, An Analysis of Particle Swarm Optimizers. PhD thesis, Department of Computer Science., 2006, University of Pretoria, Pretoria, South Africa.
[15] Mohammad Teshnehlab and Mahdi Aliyari Shoorehdeli Mojtaba Ahmadieh Khanesar, "A Novel Binary Particle Swarm Optimization," in Proceedings of the 15th Mediterranean Conference on Control and Automation, Greece, July, 2007, p.6.
[16] V. Selvi and R. Umarani, "Comparative Analysis of Ant Colony and Particle Swarm Optimization techniques," International Journal of Computer Applications, , vol. 5, no. 4, pp. 1-6, 2010.
[17] Kwang Y. Lee and Jong-Bae Park, "Application of Particle Swarm Optimization to Economic Dispatch Problem: Advantages and Disadvantages," IEEE, 2010.
[18] Ajith Abraham, and Amit Konar Swagatam Das. (2008) www.softcomputing.net. [Online]. http://www.softcomputing. net/aciis.pdf
[19] Ganesh Kumar,Salman Mohagheghi,Jean-Carlos Hernandez Yamille delValle, "Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems.," in IEEE, 2008, pp. 171-195.
[20] Y. Rahamat-Samii and Shenheng Xu, "Boundary Conditions in Particle Swarm Optimization Revisited," IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, vol. 55, no. 3, pp. 760-765, March 2007.
[21] S. M. Mikki and Ahmed A. Kishk, "Hybrid Periodic Boundary Condition for Particle Swarm Optimization," IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, vol. 55, no. 11, pp. 3251-3256, NOVEMBER 2007.
[22] James Kennedy and Tim Blackwell Riccardo Poli, "Particle swarm optimization An overview," Swarm Intelligence, vol. 1, no. 1, pp. 33–57, 2007.
[23] P. Engelbrecht F. van den Bergh, "A New Locally Convergent Particle Swarm Optimiser," IEEE Conference on Systems, Man and Cybernetics, Tunisia, 2002.
[24] Morten Løvbjerg and Thiemo Krink, "Extending Particle Swarm Optimisers with Self-Organized Criticality," IEEE Int.Congr. Evolutionary Computation, vol. 2, pp. 1588-1593, May 2002.
[25] 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.
[26] Lihua Gong and Zhengyuan Jia, "Multi-criteria Human Resource Allocation for Optimization Problems Using Multi-objective particle Swarm Optimization Algorithm," International Conference on Computer Science and Software Engineering, vol. 1, pp. 1187-1190, 2008.
[27] M. R. Pontes, C. J. A. Bastos-Filho R. A. Santana, "A Multiple Objective Particle Swarm Optimization Approach using Crowding Distance and Roulette Wheel," IEEE Ninth International Conference on Intelligent Systems Design and Applications, pp. 237-242, 2009.
[28] Tsung-Ying Sun, Sheng-Ta Hsieh, and Cheng-Wei Lin Shih-Yuan Chiu, "Cross-Searching Strategy for Multi-objective Particle Swarm Optimization,"IEEE Congress on Evolutionary Computation, pp. 3135-3141, 2007.

BIOGRAPHIES

Sachin Kumar is pursuing the M.Tech in Computer Science & Engineering from Shepherd School of Engineering & Technology in the university of Sam Higginbottom Institute of Agriculture Technology & Sciences, Allahabad. His research areas interests are Data Mining using GIS/RS, Pattern Recognition Techniques using Data Mining Tools & Techniques, Artificial Intelligence, etc

Mr. N.K. Gupta is working in the department of CSE of SHIATS from last 10 years. He has completed his UG & PG from ALLAHABAD UNIVERSITY. He is pursuing PhD from SHIATS. He is expertise in RDBMS & DATA MINING/OBJECT ORIENTED TECHNOLOGIES field. He has guided more than 40 PhD students & 50 M.Tech and published more than 20 NATIONAL & INTERNATIONAL REPUTATED JOURNALS.