Volume 5, Issue 4, 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Review on Particle Swarm Optimization

Indresh Kumar Gupta
ABV-Indian Institute of Information Technology and Management,
Gwalior, M.P., India

Abstract— Particle Swarm Optimization (PSO) is a heuristic optimization method developed in 1995 by Eberhart and Kennedy based on social practices of birds with in a flock for food finding. This technique iteratively attempting to enhance the candidate solution with respects to measure of quality for optimization problems. A lot of fundamental changes have been made to enhance the rate of convergence and solution found by PSO. The primary thought of rule of PSO and basic modification is presented. We also summarize the advantage and disadvantage to the basic modification of PSO with function.

Keywords— Particle Swarm Optimization (PSO), Inertia Weight , Convergence, Algorithm ,Exploration, Exploitation.

I. INTRODUCTION

Particle Swarm Optimization (PSO) is a swarm based search algorithm developed in 1995 by Eberhart and Kennedy taking into account the social practices of birds with in a flock for food finding [1][2] .PSO emulates the social behaviour of birds that don't have any leader in their group , will find food random, follow one of the member in the group that has closet position with the source of food . It could be implemented and activated calmly to tackle different function optimization problems; or the problem that can be adapted to function optimization problem [3]. The capital backbone of PSO is its fast convergence, which compares agreeably with abounding global optimization algorithm like genetic algorithms (G.A) simulated annealing (SA) and all other optimization algorithms. While the swarm based algorithms are more excessive due to their reliance specifically upon the function values opposed to the subsidiary data; they are however susceptible to premature convergence which is particularly the situation when there are many dimension to be optimized.

II. PARTICLE SWARM OPTIMIZATION

PSO algorithms keeps ups a swarm of particles, where each particle be a potential solution .In evolutionary computing, swarm is equivalent to a population, while a particle is equivalent to an individual. In PSO the particles called potential solution are "flown "through search space each particles knows the best solution (fitness) till now [1][2] . This value is known as p_{best} .Another best position which is recorded by the PSO algorithms is the best value, obtained so far by any particle among all particles in the neighbourhood. This value is known as g_{best} . The i^{th} particle is denoted as $X_i = (x_{i1}, x_{i2}, \dots\dots, x_{in})$ and the rate of progress position of particle is called velocity and denoted as $V_i = (v_{i1}, v_{i2}, \dots v_{in})$ particle changes its positions and velocity using the following equations .

$$V_i(k+1) = V_i(k) + c_1 * r_1\left(p_{best} - X_i(k)\right) + c_2 * r_2\left(g_{best} - X_i(k)\right) \quad (1)$$

$$X_i(k+1) = X_i(k) + V_i(k) \quad (2)$$

Where

$V_i(k)$ Denotes the velocity of particle i at iteration k.

$X_i(k)$ Denotes the position of particle i at iteration k

$V_i(k+1)$Denotes the velocity of particle i at iteration k+1

$X_i(k+1)$Denotes the position of particle i at iteration k+1

r_1 and r_2 are random number between $(0,1)$

$c_1$is cognitive acceleration coefficient and c_2 is social acceleration coefficient

$X_i(0) \sim \cup\,(X_{min}, X_{max})$

Here we consider the search space is n dimension.

Velocity component: The past speed,$V_i(k)$, which serves as a memory of the past flight bearing. Which keeps the particle from enormously altering path, and to bias closer to the current path? This factor is also known as the inertia component. The cognitive aspects, $c_1 * r_1\left(p_{best} - X_i(k)\right)$, which evaluates the performance of particle i with respect to past exhibition. The impact of this term is that particles are stepped back to their own best positions, looking like the propensity of individuals to come back to circumstances that fulfilled them most previously. The social aspects, $c_2 * r_2\left(g_{best} - X_i(k)\right)$, which evaluates the performance of particle i with respect to a group of particles. The impact of

the social component is that every particle is additionally drawn towards the best position found by the particle's neighborhood.

Termination Conditions: The last part of the PSO algorithms concerns the termination conditions, i.e. criteria used to end the iterative search process. At the point when selecting a termination condition, two imperative viewpoints must be considered

1. The termination condition should not cause the premature convergence of PSO.
2. The termination condition ought to secure against oversampling of the fitness. If a termination condition need frequent computation of fitness function, computational complexity of search process can significantly increased.

The given below stopping condition have been used:

- PSO Algorithm can be ended when a maximum number of iterations or function evaluation has been surpassed.
- Algorithm can be terminating when an acceptable solution has found. Assume that X^* be the optimum of the objective function f .At that point, this criterion will end the search process as soon as a particle X_i, is discovered such that $f(X_i) \leq |f(X^* - \epsilon)|$; that is an acceptable error has been reached. The value of threshold epsilon has to be selected take care [4][5][6].
- Terminate when no improvement is found over a number of iterations. There are distinctive routes in which improvement can be measured. For example, if the normal change in particle positions is small, the swarm can be considered to have converged. On the other hand, if the normal particle velocity over a number of iterations is pretty nearly zero, just small position upgrades are made, and the search process can be ended. Sadly, these stopping conditions present two parameters for which sensible qualities need to be found: unfortunately, these termination conditions present two parameters for which sensible values need to be found: (1) the window of iterations (or function evaluation) for which the execution is observed, and (2) a limit to show what constitutes inadmissible performance.
- Terminate when the normalized swarm radius is close to zero .when the normalized swarm radius is close to zero [7], the swarm has little potential for improvement, unless the global best is still moving.

PSO Parameters: The PSO is impacted by various control parameters, namely the dimension of problem, swarm size, acceleration coefficients, size of neighbourhood, number of iterations, and random number values that scale the commitment of the cognitive and social segments.

- Swarm size, N , i.e. total particles in the swarm: the larger the initial diversity if the more particles in the swarm - given that a decent uniform initialization plan is utilized to initialize the particles. A larger swarm permits larger parts of the search space to be secured per iteration. It is additionally the case that more particles may prompt less iterations to achieve a decent solution, contrasted with smaller swarms. It has been demonstrated in various empirical studies that the PSO can discover optimal solutions with small swarm sizes of 10 to 30 particles [8][9].
- Size of Neighborhood: The neighborhood size characterizes the extent of social collaboration within the swarm. The smaller the neighborhoods, the less collaboration happens. While smaller neighborhoods are slower in convergence, they have more reliable convergence to optimal solution. Smaller neighborhood sizes are less prone to local minima. To take the benefits of small and large neighborhood sizes, begin the search with small neighborhoods and build the neighborhood estimate relatively to the increment in number of iterations [10].This methodology guarantees an introductory high diversity with faster convergence as the particles move towards a promising search area.
- Number of iterations: The total number of iteration to achieve a decent solution is moreover problem dependent. Excessively few iterations may end the search rashly. A too large number of iterations have the result of unnecessary added computational complexity (provided that number of iterations is the only termination condition).
- Acceleration coefficient: The acceleration coefficient, c_1 and c_2, together with random values r_1 and r_2, control the stochastic impact of the cognitive and social components on the overall velocity of a particle .and c_1 communicates the amount of certainty a particle has in itself, while c_2 communicates the amount of certainty a particle has in its neighbors. With $c_1 = c_2 = 0$ particles continue flying at their current velocity until they hit a boundary of the search space. If $c_1 > 0$ and $c_2 = 0$, particle perform local search. Then again, on the off chance that $c_2 > 0$ and $c_1 = 0$, the whole swarm is pulled to a single point.

Algorithm : Particle Swarm Optimization
Input: Number of particles in swarm(N) Acceleration constants(c_1, c_2) Velocity range (V_{max}, V_{min}) Maximum Iterations(K_{max})

Acceptable Error (Acc_{err})

Output: Global Best Particle (g_{best})

Begin:
For each particle in population
Randomly initialize the particle velocity and position
End For

Set $k = 0$
While $k < K_{max}$ *or* Acc_{err} criteria is not met do
For each particle do
Calculate fitness value
If new fitness value is better than p_{best} then
p_{best} = new fitness value
End If
End For
Find the particle as g_{best} among all p_{best} .
For each particle do
Calculate particle velocity using equation (1)
change particle position using equation (2)
End For
$k = k + 1$
End while

III. BASIC MODIFICATION TO PARTICLE SWARM OPTIMIZATION

Particle Swarm optimization (PSO) with Velocity Clamping: An imperative angle that decides the effectiveness and exactness of an optimization algorithm is the exploration–exploitation trade-off. Exploration is ability of search algorithm to explore different regions of search space in order to locate a good optimum. Exploitation on the other hand, is ability of search algorithm to focus the search around a promising area in order to refine a candidate solution. A decent optimization algorithm ideally adjusts these opposing objectives. Inside the PSO, these objectives are tended by the velocity update equation. To control the global exploration of particles, velocity is clamped to stay inside boundary limitations [11]. In the event if a particle's velocity surpasses a predetermined maximum velocity, the particle's velocity is situated to the maximum velocity. Let $V_{max,j}$ signify the maximum permitted velocity in dimension j. Particle velocity is at that point balanced before the position update using ,

$$V_{ij}(k+1) = \begin{cases} v_{ij}(k+1) & if\ v_{ij}(k+1) < V_{max,j} \\ V_{max,j} & if\ v_{ij}(k+1) \geq V_{max,j} \end{cases} \quad (3)$$

Large values of $V_{max,j}$ encourage global exploration, while smaller values support local exploitation. In the event that if $V_{max,j}$ is too small, the swarm may not explore abundantly over locally good regions. Likewise, too small values for $V_{max,j}$ raise the number of time steps to reach an optimum. Besides, the swarm may become trapped in local optimum, with no way to get out. On the other hand, too large values of $V_{max,j}$danger the likelihood of missing a good region. The particles may hop over good solutions, and keep to search in fruitless regions of search space. This leaves the issue of discovering a decent value for every $V_{max,j}$ so as to adjust between (1) moving too fast or too slow (2) exploration and exploitation Generally, the $V_{max,j}$values are chosen to be a fraction of the domain of each dimension of the search space. That is

$$V_{max,j} = \delta\left(x_{max,j} - x_{min,j}\right) \quad (4)$$

here $x_{max,j}$ and $x_{min,j}$ are respectively the maximum and minimum values of the domain of x in dimension j and $\delta \epsilon (0,1]$. The estimation of δ is problem subordinate, as was found in various observational studies [12][13]. The best estimation ought to be found for every different problem utilizing observational systems, for example, cross-validation.

Particle Swarm Optimization (PSO) with Inertia Weight: The inertia weight was presented by Shi and Eberhart [14] as a system to control the exploration and exploitation capacities of the swarm, and as a system to dispose of the requirement for velocity clamping [3]. The inertia weight was effective in tending to the first objective, yet couldn't totally dispose of the requirement for velocity clamping. The inertia weight$w$, controls the force of the particle by weighing the commitment of the past velocity – fundamentally controlling the amount of memory of the past flight heading will impact the new velocity. Particles update their velocity using ,

$$V_i(k+1) = wV_i(K) + c_1 * r_1\left(p_{best} - X_i(k)\right) + c_2 * r_2\left(g_{best} - X_i(k)\right) \quad (5)$$

The estimation of $w$is amazingly essential to guarantee convergent conduct, and to ideally trade-off exploration and exploitation. For$w \geq 1$, velocity increments take place over the time steps quickening towards the maximum velocity (expecting velocity clamping is utilized), and the swarm wander. Particles neglect to alter course so as to move back towards guaranteeing regions. For $w < 1$ particles decelerate until their velocity achieve zero (depending on the estimations of the acceleration coefficients). Large values for w encourage exploration, with expanded diversity. A small w advances the local exploitation. After all, too small values dispose of the exploration capacity of the swarm. Little force is then protected from the past time step, which empowers brisk changes in direction .The smaller$w$, the more do the cognitive and social parts control position updates. Beginning usage of the inertia weight utilized a static worth for the whole search duration, for all particles for every measurement. Later usage made utilization of dynamically changing inertia values. These methodologies as a rule begin with large inertia values, which diminish over time to smaller values. In doing as such, particles are permitted to explore in the starting search steps, while favouring exploitation as time increments. Ways to dynamically changing the inertia weight can be gathered into the following categories.

Random Inertia weight was proposed by Eberhart and Shi [15] and experimentally found that this approach increases the convergence of PSO in early iterations of the algorithm. The inertia weight is taken as,

$$w = 0.5 + \frac{Rand()}{2} \tag{6}$$

This creates a number randomly fluctuating somewhere around 0.5 and 1 .O, with a mean estimation of 0.75.

Adaptive Inertia weight was proposed by clerc [16] and improves the searching capability, where the measure of change in the inertia weight is relative to the relative improvement of the swarm. The inertia weight is balanced by,

$$w(k+1) = w(0) + \left(w(n_k) - w(0)\right)\frac{e^{m_i(k)} - 1}{e^{m_i(k)} + 1} \tag{7}$$

Where relative improvement,$m_i(k)$, is calculated as

$$m_i(k) = \frac{g_{best} - current}{g_{best} + current} \tag{8}$$

Linear Decreasing Inertia weight was proposed by Xin [16] to upgrdes the efficiency and performance of PSO. It starts with large inertia weight which decreases linearly to small value .The inertia weight is balanced by

$$w(k) = \left(w(0) - w(k_{max})\right)\frac{(k_{max} - k)}{k_{max}} + w(k_{max}) \tag{9}$$

Where k_{max} represents the maximum allowable iterations. k denotes the current iteration and $w(0)$ and $w(k_{max})$are the initial inertia weight and final inertia weight respectively. It is found experimentally $w(0) = 0.9$ and $w(k_{max}) = 0.4$ produces fabulous results.

Sigmoid Increasing Inertia weight was presented by Malik [17] and experimentally found that sigmoid function has devoted in getting least fitness function while linearly Increasing Inertia weight offers commitment to quick convergence ability. So they blend sigmoid function and Linear Increasing Inertia Weight and give a sigmoid increasing inertia weight which has delivered an extraordinary change in fast convergence ability and forceful development narrowing towards the solution region. The inertia weight is balanced by,

$$w(k) = \frac{w(0) - w(k_{max})}{(1 + e^{u*(k - n*k_{max})})} + w(k_{max}) \tag{10}$$

$$u = 10^{(log(k_{max}) - 2)} \tag{11}$$

Where k_{max} represents the maximum allowable iterations. k denotes the current iteration and $w(0)$ and $w(k_{max})$are the initial inertia weight and final inertia weight respectively. Furthermore sharpness of function is adjusted by constant u and n is the constant to set the partition of sigmoid function.

Non linear Decreasing Inertia weight: where an initially large value reduces nonlinearly to a small value. Nonlinear decreasing systems permit a shorter exploration time than the linear decreasing strategies, with additional time spent on refining solution (exploiting). The accompanying nonlinear systems have been defined.

From Peram et al. [18],

$$w(k+1) = \frac{(w(k) - 0.4)(k_{max} - k)}{k_{max} + 0.4} \tag{12}$$

With $w(0) = 0.9$

From Venter and Sobieszczanski-Sobieski [19][20],

$$w(k+1) = \alpha w(k') \tag{13}$$

Where $\alpha = 0.975$ and k' is iteration when last change in inertia take place .

Chaotic Inertia weight was given by Frank [21] and Examination between chaotic inertia weight PSO and random inertia weight PSO has been carried out and found that chaotic PSO performs fabulously. The inertia weight is balanced by ,

$$w(k) = (w(0) - w(k_{max}) * \frac{k_{max} - k}{k_{max}} + w(k_{max}) * z \quad (14)$$

Where $z = 4 * z(1 - z)$ and $z \in (0,1)$.

Particle Swarm optimization (PSO) with Constriction coefficient: Clerc built up a methodology very much alike to the inertia weight to adjust the exploration–exploitation exchange trade-off. Where the velocities are constricted by a constant χ **,** referred to as constriction coefficient [22][23].

Thus velocity update equation is given as

$$V_i(k+1) = \chi \left[V_i(k) + \varphi_1(p_{best} - X_i(k)) + \varphi_2(g_{best} - X_i(k))\right] \quad (15)$$

With

$$\chi = \frac{2\mu}{2 - \emptyset - \sqrt{\emptyset(\emptyset - 4)}} \quad (16)$$

With $\emptyset = \emptyset_1 + \emptyset_2, \emptyset_1 = c_1 r_1$ and $\emptyset_2 = c_2 r_2$. Equation (16) is utilized under the restriction that $\emptyset \geq 4$ and $\mu \epsilon [0,1]$. The above equation was gotten from a formal eigenvalue examination of swarm flow [23]. The constriction methodology was produced as a natural, dynamic approach to guarantee convergence to a stable point, without the requirement for velocity clamping. The constriction coefficient, χ, assesses to a worth in the range [0, 1] which infers that the speed is lessened at every time step. The parameter, μ, in equation (16) controls the exploration and exploitation capacities of the swarm. For $\mu \approx 0$, quick convergence is acquired with neighbourhood exploitation. The swarm shows a very nearly hill-climbing conduct. Then again $\mu \approx 1$, outcomes in slow convergence with a high level of exploration. Normally, μ is situated to a steady value. Be that as it may, an initial high level of exploration with neighbourhood exploitation in the later pursuit stages can be accomplished utilizing initial value near to one, diminishing it to zero. Eberhart and Shi demonstrated experimentally that if velocity clamping and constriction are utilized together, quicker convergence rates can be acquired [24].

Table The Basic Modification of PSO

Basic Modification of PSO	Function	Advantages	Disadvantages
Velocity Clamping	Control the global exploration of swarm If particle velocity surpasses a predetermined maximum velocity ;then size of velocity is diminished so that particle stay inside the boundary limits	It diminished the velocity , thus movement of particle can be controlled	In the event that the all velocity gets to equivalent to V_{max} the particle will keep on leading searches with in a hypercube and will likely stay in optima however will not unite in the neighborhood.
Inertia Weight	Control the force of particle by weighing the commitment of past velocity	A larger value for inertia weight in the end of search will encourage the convergence ability	Attain to optimally convergence unequivocally affected by inertia weight
Constriction Coefficient	Dynamic approach to guarantee convergence to stable point	Like inertia weight	At the point when PSO converges, the settled estimations of parameters may brings about the superfluous change of particles

IV. CONCLUSIONS

The procedure of PSO algorithm in discovering optimal values follows the social practices of birds within flock for food finding. In this paper we have review on essential considered principle of PSO and its basic modification. The Basic modification reviewed in this paper controls the velocity and convergence stability of PSO. Summary is made for absolute function, advantage, disadvantage of basic modification of PSO.

REFERENCES

[1] Eberhart, R. C., & Kennedy, J. (1995, October). A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science* (Vol. 1, pp. 39-43).

[2] Kennedy, J. (2010). Particle swarm optimization. In *Encyclopedia of Machine Learning* (pp. 760-766). Springer US.

[3] Eberhart, R. C., & Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on* (Vol. 1, pp. 81-86). IEEE.

[4] Braendler, D., & Hendtlass, T. (2002). The suitability of particle swarm optimisation for training neural hardware. In *Developments in Applied Artificial Intelligence* (pp. 190-199). Springer Berlin Heidelberg.

[5] Hendtlass, T., & Randall, M. (2001, December). A survey of ant colony and particle swarm meta-heuristics and their application to discrete optimization problems. In *Proceedings of the Inaugural Workshop on Artificial Life* (pp. 15-25).

[6] Schutte, J. F., & Groenwold, A. A. (2005). A study of global optimization using particle swarms. *Journal of Global Optimization*, *31*(1), 93-108.

[7] Van Den Bergh, F. (2006). *An analysis of particle swarm optimizers* (Doctoral dissertation, University of Pretoria).

[8] Brits, R., Engelbrecht, A. P., & Van den Bergh, F. (2002, November). A niching particle swarm optimizer. In *Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning* (Vol. 2, pp. 692-696). Singapore: Orchid Country Club.

[9] Van den Bergh, F., & Engelbrecht, A. P. (2001). Effects of swarm size on cooperative particle swarm optimisers.

[10] Suganthan, P. N. (1999). Particle swarm optimiser with neighbourhood operator. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on* (Vol. 3). IEEE.

[11] Eberhart, R., Simpson, P., & Dobbins, R. (1996). *Computational intelligence PC tools*. Academic Press Professional, Inc..

[12] Omran, M., Salman, A., & Engelbrecht, A. P. (2002, November). Image classification using particle swarm optimization. In *Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning* (Vol. 1, pp. 18-22).

[13] Shi, Y., & Eberhart, R. C. (1998, January). Parameter selection in particle swarm optimization. In *Evolutionary programming VII* (pp. 591-600). Springer Berlin Heidelberg.

[14] Shi, Y., & Eberhart, R. (1998, May). A modified particle swarm optimizer. In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on (pp. 69-73). IEEE.

[15] Eberhart, R. C., & Shi, Y. (2001). Tracking and optimizing dynamic systems with particle swarms. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on* (Vol. 1, pp. 94-100). IEEE.

[16] Nikabadi, A., & Ebadzadeh, M. (2008). Particle swarm optimization algorithms with adaptive Inertia Weight: A survey of the state of the art and a Novel method. *IEEE journal of evolutionary computation*.

[17] Malik, R. F., Rahman, T. A., Hashim, S. Z. M., & Ngah, R. (2007). New particle swarm optimizer with sigmoid increasing inertia weight. *International Journal of Computer Science and Security*, *1*(2), 35-44.

[18] Peram, T., Veeramachaneni, K., & Mohan, C. K. (2003, April). Fitness-distance-ratio based particle swarm optimization. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE* (pp. 174-181). IEEE.

[19] Venter, G., & Sobieszczanski-Sobieski, J. (2004). Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization. *Structural and Multidisciplinary Optimization*, *26*(1-2), 121-131.

[20] Venter, G., & Sobieszczanski-Sobieski, J. (2003). Particle swarm optimization. *AIAA journal*, *41*(8), 1583-1589.

[21] Feng, Y., Teng, G. F., Wang, A. X., & Yao, Y. M. (2007, September). Chaotic inertia weight in particle swarm optimization. In *Innovative Computing, Information and Control, 2007. ICICIC'07. Second International Conference on* (pp. 475-475). IEEE.

[22] Onwubolu, G. C., & Clerc, M. (2004). Optimal path for automated drilling operations by a new heuristic approach using particle swarm optimization. *International Journal of Production Research*, *42*(3), 473-491.

[23] Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on*, *6*(1), 58-73.

[24] Eberhart, R. C., & Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on* (Vol. 1, pp. 84-88). IEEE.

