Paper from Proceeding of the National Conference "Science in Media 2012" Organized by YMCA University of Science and Technology, Faridabad, Haryana (India) December 3rd -4th 2012 Published by IJMRS's International Journal of Engineering Sciences, ISSN (Online): 2277-9698
www.ijmrs.com

Particle Swarm Optimization - A new optimization technique

Jyoti and Neetu Gupta, YMCA University of Science & Tech,Faridabad
G.D.Mishra, Amity University , Noida

I. INTRODUCTION

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling.

PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.

Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied.

II BACKGROUND The term "Artificial Life" (ALife) is used to describe research into human-made systems that possess some of the essential properties of life.

ALife studies how computational techniques can help when studying biological phenomena and ALife studies how biological techniques can help out with computational problems. Actually, there are already lots of computational techniques inspired by biological systems. For example, artificial neural network is a simplified model of human brain; genetic algorithm is inspired by the human evolution. Here we discuss another type of biological system - social system, more specifically, the collective behaviors of simple individuals interacting with their environment and each other. Someone called it as swarm intelligence.

The algorithm of PSO emulates from behavior of animals societies that don't have any leader in their group or swarm, such as bird flocking and fish schooling. Typically, a flock of animals that have no leaders will find food by random, follow one of the members of the group that has the closest position with a food source (potential solution). The flocks achieve their best condition simultaneously through communication among members who already have a better situation. Animal which has a better condition will inform it to its flocks and the others will move simultaneously to that place. This would happen repeatedly until the best conditions or a food source discovered. The process of PSO algorithm in finding optimal values follows the work of this animal society.

Recently, there are several modifications from original PSO. It modifies to accelerate the achieving of the best conditions. The development will provide new advantages and also the diversity of problems to be resolved. arm of particles, where particle represent a potential solution.

III THE ALGORITHM

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest.

After finding the two best values, the particle updates its velocity and positions

Let $x_i(t)$ denote the position of particle i in the search space at time t step; unless otherwise stated, t denotes discrete time steps. The position of the particle is changed by adding a velocity, $v_i(t)$ to the current position [1]:

$$x_i(t+1)= x_i(t)+ v_i(t+1) \quad (1)$$

where $v_i(t) = v_i(t-1) + c_1r_1$ (localbest(t)- $x_i(t-1)$)$+c_2r_2$ (globalbest(t)- $x_i(t-1)$) (2)

with $x_i(0) \sim U(x_{min}, x_{max})$, acceleration coefficient $c_1$and c_2 and random vector $r_1$and r_2 . Simple example of PSO, there is a function [3]:

Mln f(x) where $x(B) \leq x < x(A)$,Denote x(B)as a lower limit and x(A) as an upper limit.

IV THE PSEUDO CODE

The pseudo code of the procedure is as follows

For each particle
Initialize particle
END

Do
For each particle
Calculate fitness value
If the fitness value is better than the best fitness value (pBest) in history
set current value as the new pBest

End
Choose the particle with the best fitness value of all the particles as the gBest

For each particle
Calculate particle velocity according equation (1)
Update particle position according equation (2)
End
While maximum iterations or minimum error criteria is not attained
At the i^{th} iteration, find the two important parameters for each particle that is:

a. The best value of x_j (i) (the coordinates of particle at iteration i) and declare as $p_{best}(j)$, with the lowest value of objective function (minimization case) $f[x_j(i)]$, which found a particle j at all previous iteration. The best value $x_j(i)$ for all particles which found up to the i^{th} iteration, gbest with the value function the smallest goal / minimum among all particles for all the previous iterations,$f[x_j(i)]$, .

b. Calculate the velocity of particle *j* at iteration *i* using the following formula using formula (2):

Where c_1 and c_2, respectively, are learning rates for individual ability (cognitive) and social influence (group), and and uniformly random numbers are distributed in the interval 0 and 1. So the parameters and represent weight of memory (position) of a particle towards memory (position) of the groups (swarm). The value of $c_1$and c_2 is usually 2, so multiply c_1r_1 and c_2r_2 ensure that the particles will approach the target about half of the difference.

c. Calculate the position or coordinates of particle j at the i^{th} iteration by :

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Evaluation of the objective function value for each particle and expressed as:

$f[x_1(i)], f[x_2(i)],$---------,$f[x_n(i)]$

The last step, check whether the current solution is convergent. If the positions of all particles leading to an equal value, then this is called convergence. This iteration process continues until all particles converge to the same solution.

If the current solution is convergent, then the iteration will stop. We do not know whether the final value is the best value. Below are the stopping criteria conditions for the iteration:

- Terminate when a maximum number of iterations, or FEs, has been exceeded.
- Terminate when an acceptable solution has been found,
- Terminate when no improvement is observed over a number of iteration.
- Terminate when the normalized swarm radius is close to zero.
- Terminate when the objective function slope is approximately zero. Although the particle has stopped, we do not know whether the particle will pitch on local optima, local minima, global optima or global optima.

In the original particle swarm optimization, there has also a lack of solution, because it is very easy to move to *local optima*. In certain circumstances, where a new position of the particle equal to global best and local best then the particle will not change its position. If that particle is the global best of the entire swarm then all the other particles will tend to move in the direction of this particle. The end of result is the swarm converging prematurely to a local optimum. If the new position of the particle is pretty far from global best and local best then the velocity will change quickly and turn into a great value. This will directly affect the particle's position in the next step. For now the particle will have an updated position of great value, as a result, the particle may be out of bound of the search area.

V COMPARISONS BETWEEN GENETIC ALGORITHM AND PSO

Most of evolutionary techniques have the following procedure:

1. Random generation of an initial population
2. Reckoning of a fitness value for each subject. It will directly depend on the distance to the optimum.
3. Reproduction of the population based on fitness values.
4. If requirements are met, then stop. Otherwise go back to (2) .

From the procedure, we can learn that PSO shares many common points with GA. Both algorithms start with a group of a randomly generated population, both have fitness values to evaluate the population. Both update the population and search for the optimium with random techniques. Both systems do not guarantee success.

However, PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory, which is important to the algorithm.

Compared with genetic algorithms (GAs), the information sharing mechanism in PSO is significantly different. In GAs, chromosomes share information with each other. So the whole population moves like a one group towards an optimal area. In PSO, only gBest (or lBest) gives out the information to others. It is a one -way information sharing mechanism. The evolution only looks for the best solution. Compared with GA, all the particles tend to converge to the best solution quickly even in the local version in most cases.

VI BASIC VARIANTS OF PSO

The lacks of PSO have been reduced with a variation of PSO. Many variations have been developed to improve speed of convergence and quality of solution found by the PSO. The variation is influenced by a number of control parameters, namely the dimension of the problem, the number of particles (swarm size), acceleration coefficients (The acceleration coefficient, c_1 and c_2 together with random vector r_1 and r_2, control the stochastic influence), inertia weight, neighborhood size, number of iteration, and the random values which scale the contribution of the cognitive and social component. Below are the basic variations of particle swarm optimization.

Velocity clamping

It will control the global exploration of the particle. If the velocity of a particle exceeds the maximum allowed speed limit, it will set a maximum value of velocity $v_{max}(j)$. So that $v_{max}(j)$ indicates the maximum allowable speed for a particle in the j^{th} dimension. Speed (velocity) of the particle is adjusted using the equation [2]:

$V_{ij} = \{ v_{ij}\ if\ v_{ij} < v_{max}(j),\ v_{max}(j)$ otherwise

High value of $v_{max}(j)$ will cause global exploration, whereas lower values result in local exploration. $v_{max}(j)$ will control the movement of the particle and aspect of exploration and exploitation.

[Exploration is the ability of a search algorithm to explore different region of the search space in order to locate a good optimum. Exploitation, on the other hand, is the ability to concentrate the search around a promising area in order to refine a candidate solution]

Velocity clamping did not influence the position of the particle. This only reduces the size of the step velocity. Changes in the search direction not only can make a particle to perform a better exploration but also has negative effects and the optimum value cannot be found.

The following equation [2] is used to initialize the max and min velocity to the solution:

$v_{max,}j = \delta\ (x_{max,}j - x_{min,}j)$ (4)

$v_{min,}j = \delta\ (x_{min,}j - x_{max,}j)$ (5)

Where $x_{max,}j$ and $x_{min,}j$ are the minimum and maximum positions of the particle in the dimension. δ is a constant factor and is taken from 0 until 1. The problem is if all the velocity becomes equal to v_{max} the particle will continue to conduct searches within a hypercube and will probably remain in the optima but will not converge in the local area. Some researchers have developed velocity clamping method, such as : [2], [3]

Inertia weight

It is a mechanism to control an exploration and exploitation abilities of the swarm, and as mechanism to eliminate the need of velocity clamping. The inertia weight,, controls the momentum of the particle by weighing the contribution of the previous velocity – basically controlling how much memory of the previous flight direction will influence the new velocity. For the PSO, the velocity equation [5] changes from equation:

$V_{ij}(t+1) = wv_{ij}(t) + c_1r_{1j}(t)(y_{ij}(t) - x_{ij}(t)) + c_2r_{2j}(t)$ ($\hat{y}_j(t) - x_{ij}(t)$) (6)

A similar change is made from the- PSO. Inertia weight presenting how much the amount of memory from the previous flight direction will affect the new velocity. If $w > 1$, then the velocity will decrease with time, the particle will accelerate to maximum velocity and the swarm will be divergent. If $w < 1$, then the velocity of particle will decrease until it reaches zero. The larger value of w

will facilitate an exploration, rather small values will promote the exploitation.There are some researchers that have develop inertia weight application, such as :[4], [5], [6], [7]

Constriction Coefficient

Velocity update equation that using constriction coefficient changes to:

v_{ij} (t+1)=

$$x[v_{ij}(t) + \varphi_1(y_{ij}(t) - x_{ij}(t) + \varphi_2(\hat{y}_j(t) - x_{ij}(t)]$$

(7)

Where $x = \frac{2\kappa}{2-\varphi-\sqrt{\varphi(\varphi-4}}$

With $\varphi = \varphi_1 + \varphi_2$

Where $\varphi_1 = c_1 r_1$, $\varphi_2 = c_2 r_2$

Equation above is used under the constraints that $\varphi \geq 4 \; and \; k \in [0,1]$. The constriction approach was developed as a natural, dynamic way to ensure convergence to a stable point, without the need for velocity clamping. Condition $\varphi \geq 4 \; and \; k \in [0,1]$, of the swarm is guaranteed to convergence.

some researchers have developed constriction coefficient , such as : [7], [8].

Synchronous / Asynchronous Updates

Asynchronous is better for lbest, updates calculate the new best positions after each particle position update and have the advantage that immediate feedback is given about the best region of search space.

Synchronous Updates [9] are done separately from the particle (personal best and neighborhood bests) position updates, only given one feedback per iteration update, slower feedback and better for gbest .

Basic Variant	**Function**	**Advantages**	**Disadvantages**
Velocity Clamping	Control the global exploration of the particle Reduces the size of the step velocity, so that the particles remain in the search area, but it cannot change the search direction of the particle	VC reduces the size of the step velocity so it will control the movement of the particle	If all the velocity becomes equal to the particle will continue to conduct searches within a hypercube and will probably remain in the optima but will not converge in the local area.
Inertia Weight	Controls the momentum of the particle by weighing the contribution of the previous velocity,	A larger inertia weight in the end of search will foster the convergence ability.	Achieve optimality convergence strongly influenced by the inertia weight
Constriction Coefficient	To ensure the stable convergence of the PSO algorithm	Similar with inertia weight	when the algorithm converges, the fixed values of the parameters might cause the unnecessary fluctuation of particles
Synchronous and Asynchronous Updates	Optimization in parallel processing	Improved convergence rate	Higher throughput: More sophisticated finite element formulations Higher accuracy (mesh densities)

Source: [20]

VII MODIFICATION OF PSO

The new and modified form of PSO ensures good performances to obtain Single solution of continuous valued, nconstrained, static, single objective and optimization problem Multiple solutions or niche where large no Of individuals compete for the use of limited resources on physical environment.
Solutions that both optimizes the objective function and satisfies all contraints. If all constraints are not satisified, the algorithm has to balance the trades iff between optimal objective function value and no of constraint violated.
Solution of the real world problems that simultaneously satisfy no. of objectives, it is different from basic PSO that return only one solution.

VIII PSO IN SUPPLY CHAIN MANAGEMENT

The primary purpose of the supply chain model is controlling the inventory at different sites or stores while meeting customer service level requirements, therefore quantifying the trade-off between inventory investment and customer
service levels. Since the trade-off between inventory investment and service levels may change over time, this will request that the supply chain performance to be evaluated continuously so that the supply chain managers be able to make timely and right decisions.The physical structure of a supply chain clearly will influence its performance, and it is very important to design an efficient supply chain to facilitate the movements of
goods.
A hybrid swarm optimizer combines both binary and real valued parameters in one search. It simply operates on binary inputs with binary particle swarm algorithm and treats the continuous variables with real valued particle swarm. Binary PSO algorithm is used to take the location
decisions (whether or not to locate a facility at a given
candidate site), while the allocation decisions are obtained
by continuous PSO algorithm

REFERENCES

[1] B. Santosa, "Tutorial Particle Swarm Optimization," 2006.
[2] F. Shahzad, *et al.*, "Opposition-Based Particle Swarm Optimization with Velocity Clamping (OVCPSO), ," *Journal advances in computational Intelligent, AISC 61, pp, 339-2348,* 2009.
[3] M. Ben Ghalia, "Particle swarm optimization with an improved exploration-exploitation balance," in *Circuits and Systems, 2008. MWSCAS 2008. 51st Midwest Symposium on*, 2008, pp. 759-762.
[4] A. Chatterjee and P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," *Computers & Operations Research,* vol. 33, pp. 859-871, 2006.
[5] L. Yufeng, "Dynamic Particle Swarm Optimization Algorithm for Resolution of Overlapping Chromatograms," in *Natural Computation, 2009. ICNC '09. Fifth International Conference on*, 2009, pp. 246-250.
[6] S. Xianjun, *et al.*, "A Dynamic Adaptive Particle Swarm Optimization for Knapsack Problem," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, 2006, pp. 3183-3187.
[7] P. K. Tripathi, *et al.*, "Multi-Objective Particle Swarm Optimization with time variant inertia and acceleration coefficients," *Information Sciences,* vol. 177, pp. 5033-5049, 2007.
[8] K. T. Chaturvedi, *et al.*, "Particle swarm optimization with time varying acceleration coefficients for non-convex economic power dispatch," *International Journal of Electrical Power & Energy Systems,* vol. 31, pp. 249-257, 2009.
[9] P. Boonyaritdachochai, *et al.*, "Optimal congestion management in an electricity market using particle swarm optimization with time-varying acceleration coefficients," *Computers & Mathematics with Applications,* vol. In Press, Corrected Proof, 2010.

[10] A. Engelbrecht, "particle Swarm Optimization : Pitfalls and convergen aspect."
[11] V. Kalivarapu, *et al.*, "Synchronous parallelization of Particle Swarm Optimization with digital pheromones," *Advances in Engineering Software,* vol. 40, pp. 975-985, 2009.
[12] S. B. Akat and V. Gazi, "Decentralized asynchronous particle swarm optimization," in *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE*, 2008, pp. 1-8.
[13] V. Gazi, "Asynchronous Particle Swarm Optimization," in *Signal Processing and Communications Applications, 2007. SIU 2007. IEEE 15th*, 2007, pp. 1-4.
[14] I. Scriven, *et al.*, "Asynchronous multiple objective particle swarm optimisation in unreliable distributed environments," in *Evolutionary*

Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on, 2008, pp. 2481-2486.
[15] W. Bo, *et al.*, "Distributed Rate Allocation and Performance Optimization for Video Communication Over Mesh Networks," in *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, 2007, pp. VI - 501-VI - 504.
[16] Q. Liguo, *et al.*, "Design and Implementation of Intelligent PID Controller Based on FPGA," in *Natural Computation, 2008. ICNC '08. Fourth International Conference on*, 2008, pp. 511-515.
[17] T. Desell, *et al.*, "Robust Asynchronous Optimization for Volunteer Computing Grids," in *e-Science, 2009. e-Science '09. Fifth IEEE International Conference on*, 2009, pp. 263-270.
[18] R. Brits, *et al.*, "Solving systems of unconstrained
equations using particle swarm optimization," in *Systems, Man and Cybernetics, 2002 IEEE International Conference on*, 2002, p. 6 pp. vol.3.
[19] M. Qianzhi, *et al.*, "Mobile Robot Path Planning with Complex Constraints Based on the Second-Order Oscillating Particle Swarm Optimization Algorithm," in *Computer Science and Information Engineering, 2009 WRI World Congress on*, 2009, pp. 244-248.
[20] Dian Palupi Rini,M.Shamsuddin ,S.Yuhanaiz Particle Swarm Optimization: Technique, System and Challenges,*" International Journal of Computer Applications (0975 – 8887) Volume 14– No.1, January 2011*