Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 53 (2013) 491 – 496

Procedia Engineering

www.elsevier.com/locate/procedia

Malaysian Technical Universities Conference on Engineering & Technology 2012, MUCET 2012
Part 4 – Information And Communication Technology

An Overview of Particle Swarm Optimization Variants

Muhammad Imran[a,]* , Rathiah Hashim[a] and Noor Elaiza Abd Khalid[b]

[a]*FSKTM, University Tun Hussein Onn Malaysia*
[b]*FSKM University Teknologi MARA Malaysia*

Abstract

Particle swarm optimization (PSO) is a stochastic algorithm used for the optimization problems proposed by Kennedy [1] in 1995. It is a very good technique for the optimization problems. But still there is a drawback in the PSO is that it stuck in the local minima. To improve the performance of PSO, the researchers proposed the different variants of PSO. Some researchers try to improve it by improving initialization of the swarm. Some of them introduce the new parameters like constriction coefficient and inertia weight. Some researchers define the different method of inertia weight to improve the performance of PSO. Some researchers work on the global and local best particles by introducing the mutation operators in the PSO. In this paper, we will see the different variants of PSO with respect to initialization, inertia weight and mutation operators.

Keywords: PSO, Overview of PSO, PSO Variants, PSO and mutation Operators, PSO and Inertia Weight.

1. Introduction

PSO is a Mehta heuristic algorithm originally proposed by Kennedy and Eberhart [HYPERLINK \l "JKe95" 1]. The algorithm simulates the behavior of bird flock flying together in multi dimensional space in search of some optimum place, adjusting their movements and distances for better search]}. PSO is an evolutionary computation method similar to the Genetic Algorithm (GA). Swarms called particles are initialized randomly and then search for optimal by updating generations.PSO has two approaches: one is called cognitive and another is called social.

The algorithm mimics a particle flying in the search space and moving towards the global optimum. A particle in PSO can be defined as $P_i \in [a, b]$ where i=1, 2, 3…. D and a, b$\in R$, D is for dimensions and R is for real numbers [30]. Each particle has its own velocity and position which are randomly initialized in the start. Each particle have to maintain its positions p_{best} known as local best position and the G_{best} known as global best position among all the particles. Following equations are used to update the position and velocity of the particle.

$$V_i(t+1) = V_i(t) + C_1 * r_1\left(P_{best} - x_i(t)\right) + C_2 * r_2\left(G_{best} - x_i(t)\right) \quad (1)$$

* Corresponding author. *E-mail address:* malikimran110@gmail.com

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{2}$$

Where V_i is the velocity, X_i is the position P_{best} is the personal best position of the particle and g_{best} is the global best position for the PSO. r_1 , r_2 are two random numbers ranges [0, 1) and C_1 & C_2 are the leaning factors.

2. Original PSO Pseudo Code

To improve the performance of PSO, researches modified the PSO in different ways. The pseudo code of original PSO is:

```
Initialize the population randomly
While (Population Size)
{
Loop
Calculate fitness
If fitness value is better from the best fitness value ($p_{best}$) in history then
Update $p_{best}$ with the new $p_{best}$
End loop
Select the particle with the best fitness value from all particles as $g_{best}$
While maximum iterations or minimum error criteria is not attained
{
For each particle
Calculate particle velocity by equation (1)
Update particle position according to equation (2)
Next
}
}
```

3. Elements Used In PSO

Before working with the PSO, we have to know about the elements used in the PSO. First of all, we shall overview the brief concepts of the PSO elements.

- ***Particle---***We can define the particle as P_i ε [a, b] where i=1, 2, 3…D and a, b ε R. Here D is for dimension and R is for real numbers.
- ***Fitness Function---***Fitness Function is the function used to find the optimal solution. Usually it is an objective function.
- ***Local Best---***It is the best position of the particle among its all positions visited so for.
- ***Global Best---***The position where the best fitness is achieved among all the particles visited so for.
- ***Velocity Update---***Velocity is a vector to determine the speed and direction of the particle. Velocity is updated by the equation (I).
- ***Position Update---***All the particles try to move toward the best position for optimal fitness. Each particle in PSO updates their positions to find the global optima. Position is updated by equation (2)

Flow chart of the basic PSO

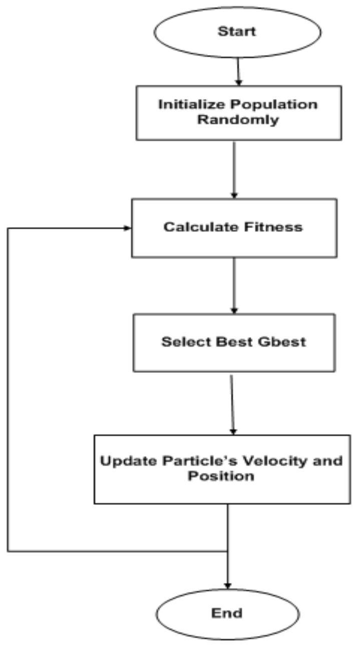
Start
Initialize Population Randomly
Calculate Fitness
Select Best Gbest
Update Particle's Velocity and Position
End

Fig. 1. Flow chart of Basic PSO.

4. Pso Applications

There are different PSO applications available in the literature like power applications, image Processing and functions optimization.

Here we will discuss more about the functions optimization.

5. Pso Variants

In 1995, Kennedy proposed PSO [1]. As it is mathematically proved so researchers are trying to improve the PSO. Therefore, PSO has different variants with different parameters like, Initialization, Inertia weight and many other techniques. The details of some PSO variants are given below.

5.1. Initialization

Initialization of the particles has an important role in the performance of PSO. If the initialization is not good then algorithm may search in unwanted area and it will be hard to search for the optimal solution. The performance of PSO heavily depends on the initialization of the swarms [3].in this section we will see study the different variants of PSO with respect to initialization.

Nguyen et al [2] et al used some low discrepancy sequence to initialize the particles. Author used Halton, Sobol, and Faure sequences to initialize the swarm. They test their proposed variants using six standard benchmark functions. They find that the performance of PSO with sobol initialization is best among all the techniques.

Jabeen et al [4] presented an opposition based PSO. Author claimed that by the social phenomena if one person is bad then his opponent is good. They generate the population and opposite population then calculate the fitness; the fitter one population is selected to run the PSO. According to author the opposite particle is defined as

P_{opi} $=a+b-p_i$ where i=1, 2, 3…. D is dimension and a, b ε R

Where D is the dimension and R is real number.

Pant et al [5] used qasi random sequence for initialization of the swarm to improve the PSO performance. Author used the vnder corput and sobol sequence for the swarm initialization. Author used four benchmark functions to perform the experiment. According to the results presented by the author the performance of VC-PSO used vendor corrupt sequence and SO-PSO used sobol sequence remain dominant on the simple PSO.

Omran [6] proposed an opposition based learning to improve the performance of PSO. They propped three variant of PSO. In one they just generate the opposite particles then calculate the fitness of both particles then select the fitter particles to run the PSO. In another variant, the lowest fitter particle was replaced with its opposite particle during each iterati-on. Eight benchmark functions ware used to test the performance of propped variants with the basic PSO. The results show that the presented variants perform well.

Chang et al [7] proposed an enhance variant of PSO which they called quasi-oppositional comprehensive learning particle swarm optimizers (QCLPSO). They used the qausi opposite numbers for the swarm initialization.

5.2. Constriction Factor

In this section, we will just overview a variant of PSO in which constriction factors has been introduced.

Clerc [8] proposed an approach to balance the exploration and exploitation by introducing a new parameter ‘χ’ called constriction factor. By clerc following equation used to update the velocity

$$v_{ij}(t+1) = \chi\left[v_{ij}(t+1) + \varphi_1\left(v_{ij}(t) - x_{ij}(t)\right) + \varphi_2(\hat{y}_{ij}(t) - x_{ij}(t))\right] \quad (3)$$

Where $\chi = 2 \div \left|4 - \varphi - \sqrt{\varphi 2 - 4\varphi}\right|$

$$\varphi = C1 + C2,\ \varphi_1 = c_1 r_1,\ \varphi_1 = c_1 r_1$$

5.3. Inertia Weight

Inertia weight is an approach like the constriction coefficient to balance the exploration-exploitation trade off. The bigger value of inertia weight encourages the exploration and smaller value of inertia weight encourages for the exploitation. Shi [9] first time introduce the inertia weight to control the exploration and exploitation. Some researcher used fix inertia weight, some used linearly decreasing and some used non-linearly decreasing inertia weight. This section discuss about the variations of basic PSO by modifying the inertia weight in different ways.

Li [10] used the exponent decreasing inertia weight. To balance the local and global search abilities the author presented the exponent decreasing inertia weight as

$$w = (w_{ini} - w_{end} - d_1)\exp\left(\frac{1}{1+d_2 t/t_{max}}\right) \quad (4)$$

Where t_{max} shows the max iteration, t shows the t^{th} iteration, w_{ini} shows the actual inertia weight, w_{end} shows the inertia weight value when the algorithm process run the max iterations, d_1 and d_2 is a factor to control w between $w_{ini}\ and\ w_{end}$. Chongpeng et al [11] proposed a variant of PSO with non-linearly decreasing inertia weight as

$$w_0 = w_{end} + (w_{end} - w_{start}) \times (1 - (t/t_{max})^{k_1})^{k_2} \quad (5)$$

Where k_1, k_2 are two natural numbers, w_{start} is the initial inertia weight, w_{end} is the final value of weighting coefficient, t_{max} is the maximum number of iteration and t is current iteration. Value of $k_1>1$ and $K_2=1$. Yuhui [12] used fuzzy set of rules to adjust inertia weight dynamically

Zhang et al [13] generate a uniformly distributed random number between [0, 1]. Author claimed that the performance of proposed inertia weight is much better than the linearly decreasing inertia weight. Moreover they said that through their technique two draw backs of linearly inertia weight can be overcome. One the dependency of inertia weight on the maximum number of iteration the another one is avoiding the lacks of local search ability in the start and global search ability at the end.

Wei et al [14] introduce a new version of inertia weight; inertia weight changed dynamically based on speed and accumulation factors.

Pant et al [15] used Gaussian distribution to set the inertia weight. Following equation used to set the inertia weight

$$w = abs(rand)/2 \tag{6}$$

Where rand is random number generated by Gaussian distribution.

Xuedan Liu et al [16] used following equation to set the inertia weight

$$w(t) = .9 - \left(\frac{t}{max_t}\right) \times 0.5 \tag{7}$$

5.4. Mutation Operators

To improve the performance of PSO and to escape it from the local minima, the researches proposed different variants of PSO with mutation operators. Some researchers mutate the global best particle and some mutate the local best particle with different techniques to prevent the PSO for stagnation in local minima. Here we will discuss some of the PSO variants with respect to mutation operators.

Wang et al [17] proposed a variant of PSO with Cauchy mutation. Author mutates the global best particle then compares its fitness with the original particle fitness, the fitter one is selected.

Following equation used to mutate the particle.

$$gbest_i(i) = gbest_i(i) + W(i) * N(x_{min}, x_{max}) \tag{8}$$

Where N is a cauchy distributed function with scale parameter t=1, $N(x_{min}, x_{max})$ is a random number between (x_{min}, x_{max}) of defined domain of test function and

$$W(i) = \left(\sum_{j=1}^{popsize} V[j]\,[i]\right)/popsize$$

Where $V[j][i]$ is the i^{th} velocity vector of j^{th} particle in the population, popsize used for population.

Wang et al [18] proposed another variant of PSO in which Cauchy mutation is used with opposition based PSO. Pant et al [19] presented a new version of PSO with adaptive mutation. They proposed two versions of PSO AMPSO1 and AMPSO2, in one they mutate the global best particle while in another local best particle was mutated. Following equation used to mutate the particle

$$g_{best} = g_{best} + \sigma' * Betarand() \tag{9}$$

Where $\sigma' = \sigma * \exp(\tau N(0,1) + \tau' N_j(0,1))$, N(0,1) is a normally distributed function with mean zero and standard deviation one, $N_j(0,1)$ is another random number generated for each value of j, τ and τ' are set as $\frac{1}{\sqrt{2n}}$ and $\frac{1}{\sqrt{2\sqrt{n}}}$ respectively and value of σ is originally set as 3. Betarand () is a random number generated by beta distribution with parameter less than 1. Pant et al [20] presented another extension of PSO using sobol mutation. They proposed two version of PSO: first one is SM-PSO1 and another one is SM-PSO2. In the first one, they mutate the best particle in the swarm. But in the second one, they work on the worst particle in the swarm. The proposed operator was defined as

$$SM = R_1 + (R_2)/lnR_1 \tag{10}$$

Where R_1 and R_2 are random numbers generated by sobol sequence.

Wu et al [23] proposed a variant of PSO by using power distribution. Author applies the power mutation on the global best particle and then check the fitness of both particles original and mutated one and select the fitter one.

Imran et al [22] proposed another power mutation operator for opposition based PSO. In their proposed technique, they initialize the PSO with opposite swarms, and then apply the power mutation on global best particle. Author claimed that in this way two times performance of PSO improved; at the initialization and also by mutating global best to prevent PSO from stagnation.

Imran et al [23] presented another variant of PSO by introducing the student T mutation. Author used the student T distribution to mutate the global best particle. They claimed that their work has performance over Cauchy mutation and adaptive mutation.

Chen [24] proposed another variant of PSO with mutation operator. In this study, author first generate the mutant particle on the basis of probability, and then check the fitness of both particle and mutant particle, then select fitter one.

6.0. Conclusion

From above study, we can to know that after proposal of PSO many researchers are working to improve its performance. In original PSO, there was no inertia weight but to improve the performance researchers introduced the inertia weight. Then they tried to improve the performance by trying the different initialization methods. Researchers also work on the global best particle to escape it from the local minima. For this purpose they introduce the different mutation operators to improve the performance of PSO.

References

[1] J. Kennedy and R. Eberhart. "Particle Swarm Optimization," In Proceedings of IEEE International Conference on Neural Networks, 1995, PP.1942-1948.

[2] N. Q. Uy, N. X. Hoai, R. McKay and P. M. Tuan. "Initializing PSO with randomized low-discrepancy sequences: the comparative results," In Proceedings of the IEEE Congress on Evolutionary Computation, 2007, pp 1985-1992.

[3] A.P. Engelbrechr, Fundamentals of Computational Swarm Intelligence, John Wiley & Sons, 2005.

[4] H. Jabeen, Z. Jalil and A.R Baig, "Opposition Based Initialization in Particle Swarm Optimization," in Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, New York, NY, USA, 2009, pp. 2047- 2052.

[5] M. Pant, R. Thangaraj, C. Grosan, and A. Abraham, "Improved Particle Swarm Optimization with Low-Discrepancy Sequences," in IEEE Cong. on Evolutionary, Hong Kong , 2008, pp. 3011-3018.

[6] M. G. H. Omran and S.al-Sharhan. "Using Opposition-based learning to improve the Performance of Particle Swarm Optimization." IEEE Swarm Intelligence Symposium, 2008, pp 1 – 6.

[7] C. Zhang et al, "A Novel Swarm Model With Quasi-Oppositional Particle," International Forum on Information Technology and Applications, 2009, pp 325 – 330.

[8] Clerc M and Kennedy J. "The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space," IEEE Transactions on Evolutionary Computation, vol. 6, 2002 ,pp 58–73.

[9] Y. Shi and R. Eberhart., "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.

[10] H-R LI and Y-L Gao., "Particle swarm optimization algorithm with exponent decreasing inertia weight and stochastic mutation," in Second International Conference on Information and Computing Science, Manchester , 2009, pp. 66-69.

[11] Huang Chongpeng, Zhang Yuling, Jiang Dingguo and Xu Baoguo, "On Some Non-linear Decreasing Inertia Weight Strategies in Particle Swarm Optimization*," in Proceedings of the 26th Chinese Control Conference, Zhangjiajie, Hunan, China, 2007, pp. 570-753.

[12] Y. Shi and R. C. Eberhart, "Fuzzy Adaptive particle Swarm Optimization," in Proceedings of the IEEE Congress on Evolutionary Computation, Seoul , South Korea, 2001, pp. 101-106.

[13] L. Zhang, H. Yu, and S. Hu, "A new approach to improve particle swarm optimization," in Proceedings of the 2003 international conference on Genetic and evolutionary computation, 2003, pp. 134-139.

[14] J. Wei and Y. Wang.. "A Dynamical Particle Swarm Algorithm with Dimension Mutation," IJCSNS International Journal of Computer Science and Network Security, Vol. 6, pp.221-224, July 2006.

[15] M. Pant and T. Thangaraj, V.P. Singh, "Particle Swarm Optimization Using Gaussian Inertia Weight," in International Conference on Computational Intelligence and Multimedia Applications, Sivakasi, Tamil Nadu , 2007, pp. 97-102.

[16] X. Liu et al, "Particle Swarm Optimization with Dynamic Inertia Weight and Mutation," in Third International Conference on Genetic and Evolutionary Computing, Guilin, 2009, pp. 620-623.

[17] H, Wang et al. "A Hybrid Particle Swarm Algorithm with Cauchy Mutation," Proceeding of IEEE Swarm Intelligence Symposium, 2007, pp 356 – 360

[18] H. Wang et al. "Opposition-based Particle Swarm Algorithm with Cauchy Mutation," IEEE Congress on Evolutionary Computation, 2007, pp 4750 – 4756.

[19] M. Pant, R. Thangaraj, and A. Abraham , "Particle Swarm Optimization Using Adaptive Mutation," in 19th International Conference on Database and Expert Systems, Washington, DC, USA, 2008, pp. 519-523.

[20] M. Pant, R. Thangaraj1, V.P Singhand and A. Abraham ", Particle Swarm Optimization Using Sobol Mutation," in First International Conference on Emerging Trends in Engineering and Technology, Nagpur, Maharashtra , 2008, pp. 367-372.

[21] Xiaoling Wu, Xiaojuan Zhao ", Particle swarm optimization based on power mutationISECS International Colloquium on Computing, Communication, Control, and Management, Sanya ,2009, pp. 464 - 467

[22] M. Imran, H.Jabeen, M. ahmad, Q. ababs, w.Bangyal and Q. Ababs ", Opposition Based PSO and Mutation Operators(OPSO with Power Mutation) ," 2nd International Conference on Education Technology and Computer, Shanghai, 2010,pp.V4-506 -508.

[23] M. Imran, Z. Manzoor, S. Ali and Q. Ababs ", Modified Particle Swarm Optimization with Student T Mutation ," International Conference on Computer Networks and Information Technology, Abbottabad, 2011,pp.283 - 286.

[24] L.Chen ",Particle Swarm Optimization with a Novel Mutation Operator," International Conference on Mechatronic Science, Electric Engineering and Computer, Jilin, 2011, pp. 970 – 973.