Local and Global Search Based PSO Algorithm

Yanxia Sun[1,*], Zenghui Wang[2], and Barend Jacobus van Wyk[1]

[1] Department of Electrical Engineering, Tshwane University of Technology, Pretoria 0001, South Africa
[2] Department of Electrical and Mining Engineering, University of South Africa, Florida 1710, South Africa
{sunyanxia,wangzengh,vanwykb}@gmail.com

Abstract. In this paper, a new algorithm for particle swarm optimisation (PSO) is proposed. In this algorithm, the particles are divided into two groups. The two groups have different focuses when all the particles are searching the problem space. The first group of particles will search the area around the best experience of their neighbours. The particles in the second group are influenced by the best experience of their neighbors and the individual best experience, which is the same as the standard PSO. Simulation results and comparisons with the standard PSO 2007 demonstrate that the proposed algorithm effectively enhances searching efficiency and improves the quality of searching.

Keywords: Local search, global search, particle swarm optimisation.

1 Introduction of PSO

PSO is an evolutionary computation technique developed by Kennedy and Eberhart [1] in 1995: it is a population-based optimisation technique, inspired by the motion of bird's flocking, or fish schooling. The particle swarms are social organizations whose overall behavior relies on some sort of communication amongst members, and cooperation. All members obey a set of simple rules that model the communication within the flock, between the flocks and the environment. Each solution is a "bird" in the flock and is referred to as a "particle". PSO is not largely affected by the size and nonlinearity of the problem, and can converge to the optimal solution in many problems [2-5] where most analytical methods fail to converge. It can, therefore, be effectively applied to different optimisation problems.

The standard particle swarm algorithm works by iteratively searching in a region and is concerned with the best previous success of each particle, the best previous success of the particle swarm as a whole, the current position and the velocity of each particle [4]. The particle searches the domain of the problem, according to

$$V_i(t+1) = \omega V_i(t) + c_1 R_1 (P_i - X_i(t)) + c_2 R_2 (P_g - X_i(t)), \quad (1)$$

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

* Corresponding author.

Y. Tan, Y. Shi, and H. Mo (Eds.): ICSI 2013, Part I, LNCS 7928, pp. 129–136, 2013.

where $V_i = \left[v_i^1, v_i^2, \cdots v_i^n \right]$ is the velocity of particle i; $X_i = \left[x_i^1, x_i^2, \cdots x_i^n \right]$ represents the position of particle i; P_i represents the best previous position of particle i (indicating the best discoveries or previous experience of particle i); P_g represents the best previous position among all particles (indicating the best discovery or previous experience of the social swarm); ω is the inertia weight that controls the impact of the previous velocity of the particle on its current velocity and is sometimes adaptive. R_1 and R_2 are two random weights whose components r_1^j and r_2^j $(j = 1, 2 \cdots, n,)$ are chosen uniformly within the interval $[0,1]$ which might not guarantee the convergence of the particle trajectory; c_1 and c_2 are the positive constant parameters. Generally the value of each component in V_i should be clamped to the range $[-v_{\max}, v_{\max}]$ to control excessive roaming of particles outside the search space.

Among these parameters, the inertia weight ω plays an important role and affects the global and local search ability of PSOs. If the value of ω is too big, the global search ability of PSO will be improved, but its local search ability will not be adequate. Otherwise, if the value of ω is small, the global search ability will decrease and the particles easily fall in premature. Some parameters of adaptive PSOs has been proposed but these usually change the inertia weight: ω is large at the beginning of the search procedure and ω decreases as time increased [7, 13]. However, there is a similar problem with the fixed inertia weight method: 1) at the beginning, the local search ability is not effective as ω is big; while 2) the global search ability is not satisfactory at the end of the search procedure as ω becomes small. To balance the local search and global search ability at the same time, a new particle swarm optimisation algorithm is proposed which can perform the local and global search es simultaneously.

In the proposed algorithm, the particles are divided into two groups. The velocity of the first group of particles is only influenced by the best experience of its neighbors. And the velocity of the second group is influenced by both the best experience of its neighbors and its own best experience. The rest of this paper is arranged as follows: In Section 2, the proposed algorithm is described. Section 3 describes the problems used to evaluate the new algorithm and the results are obtained. Finally, the concluding remarks appear in Section 4.

2 Local and Global Search Based PSO Algorithm

Referring to equation (1), the right side consists of three parts: the first is the previous velocity of the particle; the second and third are those parts contributing to the change in the velocity of a particle. As explained in [7], without these two parts, the particles will keep on flying at the current speed in the same direction until they hit the boundary. PSO will not find an acceptable solution unless there are acceptable solutions on their flying trajectories. But this is a rare case. On the other hand, referring to equation (1) without the first part, the flying particles' velocities are only

determined by their current positions and their best positions in its history. At the same time, each other particle will be flying toward its weighted centroid of its own best position and the global best position of the population [8]. Some authors have suggested adjustments to the parameters of the PSO algorithm: adding a random component to the inertia weight [9, 10], using a secondary PSO to find the optimal parameters of a primary PSO [11], and adaptive critics [12]. From our literature study and simulation experience, the optimum is often found near the global best experience in various optimisation problems. To help the particles to enhance searching the region around the global best experience, the first group particles are separated from the whole set of particles to search the area around the global best experience. Then equations (1) and (2) will be altered to

$$V_i(t+1) = 0.5 \times \omega \times V_i(t) + c_1 R_1 (P_g - X_i(t)) + c_2 R_2 (P_g - X_i(t)), \tag{3}$$

$$X_i(t+1) = X_i(t) + V_i(t+1). \tag{4}$$

As can be seen from equation (3), the particles will focus on searching the area around the best experience among their neighbors.

The particles in the second group will continue to the search the global experience of the swarm and its own best experience according to equations (1) and (2), which are the same as the standard PSO.

The following procedure can be used for implementing the proposed particle swarm algorithm:

1) Initialize the swarm, assign a random position in the problem hyperspace to each particle and calculate the fitness function which is yielded by the optimisation problem whose variables are corresponding to the elements of particle position coordinates.

2) The particles in the first group search the area according to equations (3) and (4). Meanwhile, those in the second group search the area according to equations (1) and (2).

3) Evaluate the fitness function for each particle.

4) For each individual particle, compare the particle's fitness value with its previous best fitness value. For each individual particle, compare the particle's fitness value with its previous best fitness value. If the current value X_i is better than the previous best value P_i, then set P_i as X_i.

5) Repeat steps 2) - 4) until the criterion for stopping is met (e.g., maximum number of iterations or a sufficiently good fitness value).

3 Numerical Simulation

To demonstrate the efficiency of the proposed technique, six well-known benchmarks are used to compare the proposed method and the standard PSO 2007 (Matlab version compiled in 2011) [14]. These six optimisation problems were used as shown in Table 1. Their parameters are given in Table 2. These six are famous test functions for

minimization methods; each of them has several local minima. In the numerical simulation of the proposed LGPSO method and standard PSO, the particle swarm population size is set floor($10+2\sqrt{D}$). Here D is the dimension of the optimisation problems and the function floor(A) rounds the elements of float number A to the nearest integers less than or equal to A. The rest of the parameters are as follows: inertia weight $\omega = \frac{1}{(2\log 2)} \approx 0.7213$, learning rates $c_1 = c_2 = 0.5 + \log 2$, and velocity Vmax set to the dynamic range of the particle in each dimension. The topology of LGPSO is the same as the standard PSO 2007 (SPSO 2011) [14]. It should be noted that the initial variables are set random float numbers in the range [0, 1] to check the effect of big search range. The maximum number of function evaluations is 2000 for these two methods with 100 independent runs. The optimisation statistical analysis of these two algorithms is reported in Table 3. The evolutionary curves of LGPSO and the standard PSO 2007 are depicted in Figures 1-6.

Table 1. Functions used to test the effects of the LGPSO method

Problem	Objective functions
Rosenbrock	$f(x) = \sum_{i=1}^{D} (100(x_{i+1} - x_i^2)) + (x_i - 1)^2)$
Ackley	$f(x) = 20 + e - 20e^{-\frac{1}{5}\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}} - e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)}$
Griewank	$f(x) = \frac{1}{4000}\sum_{i=1}^{D}(x_i - 100)^2 - \prod_{i=1}^{D}\cos(\frac{(x_i - 100)}{\sqrt{i}}) + 1$
Salomon	$f(x) = \cos(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2} + 1$
Rotated-hyper-ellipsoid	$f(x) = \sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2$
Quartic function	$f(x) = \sum_{i=1}^{D} i x_i^4 + rand()$
Alpine	$f(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1 x_i \rvert$
Levy	$f(x) = \sin^2(\pi x_1) + \left(\begin{array}{l} \sum_{i=1}^{D-1} (x_i - 1)^2 \left(1 + 10\sin^2(\pi x_i + 1)\right) \\ +(y_{D-1} - 1)^2 (1 + \sin^2(2\pi x_{D-1})) \end{array} \right)$

Table 2. Functions parameters for the test problems

Problem	Dimension	Search range	Initial range
Rosenbrock	30	±500	[0, 1]
Ackley	30	±500	[0, 1]
Griewank	30	±500	[0, 1]
Salomon	30	±500	[0, 1]
Rotated-hyper-ellipsoid	30	±500	[0, 1]
Quartic function	30	±500	[0, 1]
Alpine	30	±500	[0, 1]
Levy	30	±500	[0, 1]

Table 3. Comparison between standard PSO 2007 and LGPSO

Problem	Method	best	Mean	Std.dev	Worst
Rosenbrock	Standard PSO 2007	122.3422	222.7063	31.1602	343.6297
Rosenbrock	LGPSO	122.0898	183.5776	24.3594	261.8783
Ackley	Standard PSO 2007	1.8158	2.3669	0.2861	3.0259
Ackley	LGPSO	1.2924	2.0563	0.3337	2.9711
Griewank	Standard PSO 2007	0.0411	0.0788	0.0226	0.1827
Griewank	LGPSO	0.0247	0.0584	0.0172	0.1110
Salomon	Standard PSO 2007	0.2999	0.2999	1.0235e-004	0.3005
Salomon	LGPSO	0.2999	0.2999	6.9229e-006	0.2999
Rotated hyper-ellipsoid	Standard PSO 2007	38.1115	144.2199	67.4747	432.1835
Rotated hyper-ellipsoid	LGPSO	12.1226	48.0499	30.0913	143.8851
Quartic function	Standard PSO 2007	1.8093	5.8692	2.5883	15.6572
Quartic function	LGPSO	1.3892	3.7318	1.5546	8.2722
Alpine function	Standard PSO 2007	0.9011	1.9667	0.6130	4.0312
Alpine function	LGPSO	0.4702	1.4589	0.5381	3.5732
Levy function	Standard PSO 2007	0.4363	0.7213	0.1312	1.1353
Levy function	LGPSO	0.2605	0.5791	0.1267	0.8630

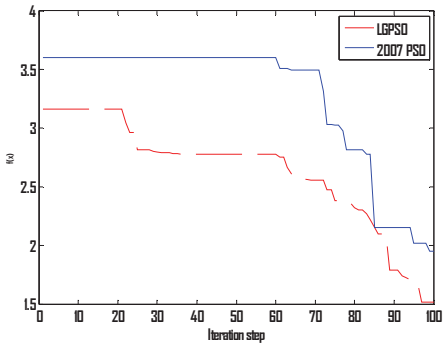

Fig. 1. Comparison for Rosenbrock function

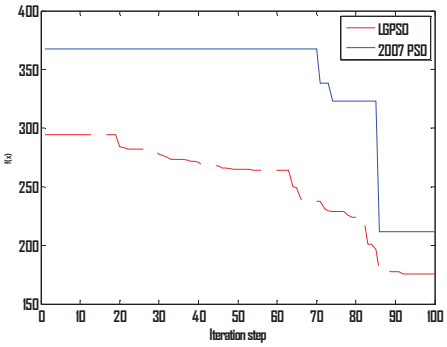

Fig. 2. Comparison for Ackley function

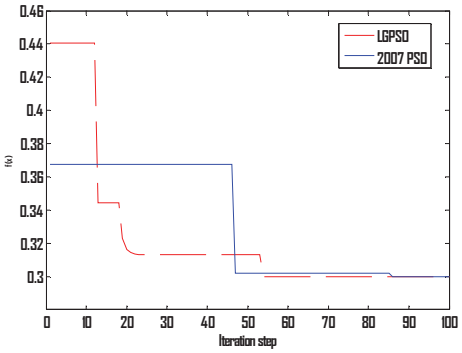

Fig. 3. Comparison for Griewank function

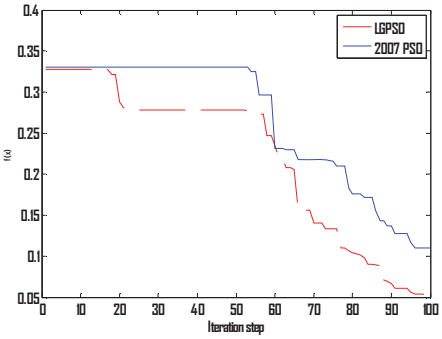

Fig. 4. Comparison for Salomon function

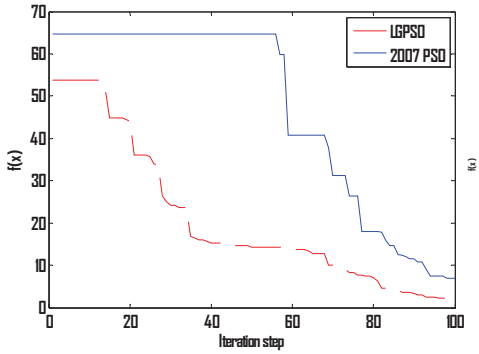

Fig. 5. Comparison for Rotated hyper-ellipsoid function

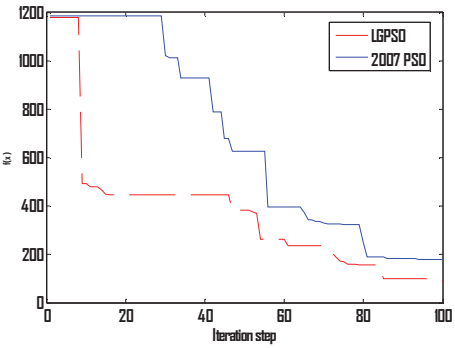

Fig. 6. Comparison for Quartic function

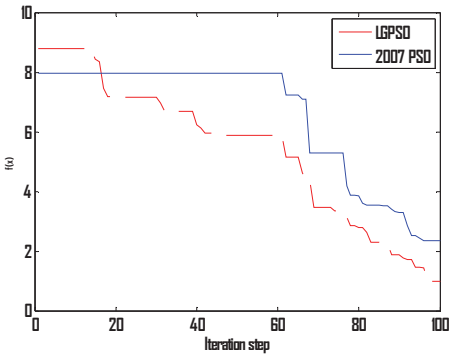

Fig. 7. Comparison for Alpine function

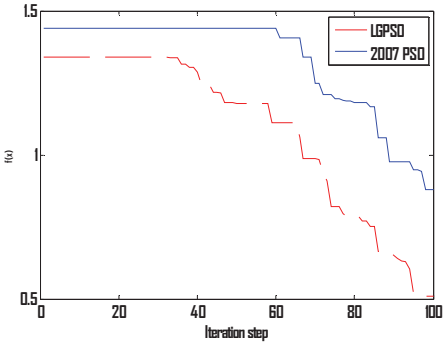

Fig. 8. Comparison for Levy function

As can be seen from Table 3, for the test functions, the best, mean, standard deviation and worst results obtained by the LGPSO are better than the results gained from the standard PSO 2007. The optimization performance of the proposed method is also more stable than the standard PSO 2007 according to the statistical analysis of mean and standard deviation. From Figures 1-8, the optimisation performance is better when the procedure begins, as the local search is added into the algorithms. The simulation results obtained by the LGPSO are better than the results from the standard PSO 2007, which means the final solutions obtained from the LGPSO are more closely focused on the best solution than those from the standard PSO 2007.

4 Conclusion

In this paper, a local and global search based particle swarm optimisation (LGPSO) method was proposed to improve the optimisation performance of the PSO. In this new model, the first group of particles focused on the search around the global best experience while the second group particles are influenced by both the best experience of their group and their own best experience. The simulations showed that the proposed method can achieve good optimisation performance no matter whether at the beginning or at the end of the search period. Moreover, the complexity of the proposed algorithm is not increased over that of the Standard PSO 2007 while the performance of the proposed FCPSO is more stable and more accurate than the Standard PSO 2007.

Acknowledgements. This work was supported by China/South Africa Research Cooperation Programme (No. 78673 & CS06-L02), South African National Research Foundation Incentive Grant (No. 81705), SDUST Research Fund (No. 2010KYTD101) and Key Scientific Support Program of Qingdao City (No. 11-2-3-51-nsh).

The authors thank the Matlab version codes of standard PSO 2007 compiled by Mahamed Omran in 2011, which are available at the Particle Swarm Central website [14].

References

1. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimisation. In: Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948 (1995)
2. Hu, X., Shi, Y., Eberhart, R.: Recent Advances in Particle Swarm. In: Congress on Evolutionary Computation, pp. 90–97. IEEE Service Center, Piscataway (2004)
3. Huang, C.M., Huang, C.J., Wang, M.L.: A Particle Swarm Optimisation to Identifying the ARMAX Model for Short term Load Forecasting. IEEE Transactions on Power Systems 20, 1126–1133 (2005)
4. Clerc, M.: Particle Swarm Optimisation. ISTE Publishing Company (2006)
5. Nedjah, N., Mourelle, L.D.M.: Systems Engineering Using Particle Swarm Optimisation. Nova Science Publishers (2007)
6. Aihara, K., Takabe, T., Toyoda, M.: Chaotic Neural Networks. Physics Letter A 144(6-7), 333–340 (1990)
7. Shi, Y., Eberhart, R.: A Modified Particle Swarm Optimiser. In: IEEE International Conference on Evolutionary Computation, pp. 69–73. IEEE Press, Piscataway (1998)
8. Zhang, W.J., Xie, X.F.: DEPSO: Hybrid Particle Swarm with Differential Evolution Operator. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Washington DC, USA, pp. 3816–3821 (2003)
9. Mohagheghi, S., Del Valle, Y., Venayagamoorthy, G., Harley, R.: A Comparison of PSO and Back Propagation for Training RBF Neural Networks for Identification of a Power System with STATCO. In: Proceedings of IEEE Swarm Intelligence Symposium, pp. 381–384 (June 2005)
10. del Valle, Y., Venayagamoorthy, G.K., Mohagheghi, S., Hernandez, J., Harley, R.G.: Particle Swarm Optimisation: Basic Concepts, Variants and Applications in Power Systems. IEEE Transactions On Evolutionary Computation 12(2), 171–195 (2008)
11. Doctor, S., Venayagamoorthy, G., Gudise, V.: Optimal PSO for Collective Robotic sSearch Applications. In: Proceeding IEEE Congress on Evolutionary Computation, Portland, Oregon, USA, pp. 1390–1395 (2004)
12. Venayagamoorthy, G.: Adaptive Critics for Dynamic Particle Swarm Optimisation. In: Proceedings of IEEE International Symposium on Intelligence Control, Taipei, Taiwan, pp. 380–384 (September 2004)
13. Liu, B., Wang, L., Jin, Y.H., Tang, F., Huang, D.X.: Improved Particle Swarm Optimisation Combined with chaos. Chaos, Solitons and Fractals 25, 1261–1271 (2005)
14. Kennedy, J., Clerc, M., et al.: Particle Swarm Central (2012), http://www.particleswarm.info/Programs.html