A novel particle swarm algorithm for multi-objective optimisation problem

Jiande Zhang*, Chenrong Huang and Jinbao Xu

Department of Computer Engineering,
Nanjing Institute of Technology,
Nanjing 211167, Jiangsu Province, China
E-mail: zhangjiandexy@gmail.com
E-mail: huangcr@njit.edu.cn
E-mail: xujb@njit.edu.cn
*Corresponding author

Jingui Lu

Department of Mechanical and Power Engineering,
Nanjing University of Technology,
Nanjing 210009, Jiangsu Province, China
E-mail: lujg@njut.edu.cn

Abstract: To maintain the diversity and convergence of Pareto optimal solutions for multi-objective problem, an improved particle swarm optimisation algorithm based on dynamical changed inertia weight is proposed to improve algorithm's ability of exploitation and exploration. By this method, if a particle finds a better solution then more energy is given onto the current velocity to speed up exploitation, and vice versa. The computer simulations for three well-known benchmark functions taken from the multi-objective optimisation literature are used to evaluate the performance of the proposed approach. Numerical experiments have been performed to evaluate the efficiency of the algorithm.

Keywords: multi-objective optimisation; particle swarm optimisation; dynamical inertia weight; adaptive parameter.

Reference to this paper should be made as follows: Zhang, J., Huang, C., Xu, J. and Lu, J. (2013) 'A novel particle swarm algorithm for multi-objective optimisation problem', *Int. J. Modelling, Identification and Control*, Vol. 18, No. 4, pp.380–386.

Biographical notes: Jiande Zhang received his BEng in Computer Science Technology from Nanjing University of Technology, Jiangsu Province, China in 2005. He received his MSc in Computer Application Technology from the same university in 2008. Three years later, he received his PhD in Mechanical Engineering from Nanjing University of Technology in 2011. Currently, he is a Lecturer with the Department of Computer Engineering, Nanjing Institute of Technology, Jiangsu Province, China. His main research interests include intelligent optimisation algorithm, image measurement and pattern recognition.

Chenrong Huang is a Professor at the College of Computer Engineering in Nanjing Institute of Technology. She received her MS in 1983. She received her BS from Hehai University, Nanjing, Jiangsu Province, China in 1988. She received her PhD from Nanjing University of Science and Technology, Nanjing, China in 2005. Her current research interests include image processing and artificial intelligent technology.

Jinbao Xu is an Associate Professor at the College of Computer Engineering in Nanjing Institute of Technology. He received his MS from Zhejiang University, Hangzhou, Zhejiang Province, China in 1992. He received his BS from Nanjing University of Science and Technology, Nanjing, Jiangsu Province, China in 2008. His current research interests include intelligent optimisation algorithm and software technology.

Jingui Lu is a Professor at CAD Centre, Nanjing University of Technology, Nanjing, China. He received his PhD from Huazhong University of Science and Technology, Wuhan, China in 1994. He received his MS in 1990. He received his BS from China University of Mining and Technology, Xuzhou, China in 1987. He was a Postdoctoral Researcher at Tokyo Institute of Technology from 1998 to 1999. He was a Visiting Professor of Utah State University in 2007. He is a senior member of the Society of Mechanical Engineer of China. His research areas and expertise include computer-aided design and a variety of computational intelligence techniques such as genetic algorithm, neural network, and decision tree.

1 Introduction

Multi-objective optimisation is an important research topic for both scientists and engineers. Over the past decades, a large amount of studies had been focused on multi-objective optimisation problem (MOOP) and had obtained a lot of achievement. The use of evolutionary algorithms for MOOP has significantly grown in the last few years. This gives rise to a wide variety of new algorithms (Shinn and Shinn, 2006). Particle swarm optimisation (PSO) is one of the evolutionary computation techniques. Kennedy and Eberhart, inspired by the choreography of a bird flock, proposed PSO algorithm (Kennedy and Eberhart, 1995). A PSO consists of a population of particles, which on the contrary to evolutionary algorithms, survive up to the last generation (Mostaghim and Teich, 2003). Particles in the swarm search the variable space by moving with a special speed toward the best position using their experience from the past generations and communication among particles. PSO is characterised by its simplicity and straightforward applicability, and it has proved to be efficient for a lot of problems in science and engineering.

In this context, many research groups in different countries have done some interacting and correlated works on multi-objective particle swarm optimisation (MOPSO). Coello and Lechuga (2002) proposed a grid method, in which the objective space is divided into many small hypercubes, and a fitness value is assigned to each hypercube depending on the number of elite particles that lie in it. It is possible that a particle does not select a suitable guide as its local guide. Parsopoulos and Vrahatis (2002) use a weighted aggregate approach and vector evaluated PSO dealing with on a number of two dimensional problems. The main problem of this method is the formulation purely for two-dimension problems. Hu and Eberhart (2002) use a dynamic neighborhood strategy to select the global best. Since the fixed objective must be selected firstly, a priori knowledge about the objective functions must be known. Mostaghim and Teich (2004) proposed a sigma method for finding the suitable global best for each particle. The sigma MOPSO method fixes the size of the archive to a certain amount using the idea of sigma dominance. Cagnina and Esquivel (2005) presented a hybrid PSO approach, including elitist policy, a mutation operator and a grid which is used as a geographical location over objective function space. Wang et al. (2006) discussed a new technique for multi-objective PSO based on fitness sharing. Wei and Wang (2007) converted the MOOP into the constrained optimisation problem. Wickramasinghe and Li (2007) proposed a new approach in selecting leaders for the particles to follow, which in-turn will guide the algorithm towards the Pareto optimal front. The algorithm uses a Differential Evolution operator to create the leaders. Ganguly et al. (2010) presented a two stages multi-objective optimisation algorithm. In the first stage, a contingency-based multi-objective planning is used to optimise the number of feeders and their routes, and the number and location of the sectionalising switches. In the second stage, the optimum sitting and sizing of the DG units is determined for the networks obtained in the first stage by another multi-objective optimisation. The improved hybrid PSO was applied for the optimisation problem of image segmentation (Zhang et al., 2011). The PSO was also found the application in the optimisation problem of multi-objective differential evolution algorithm based on the non-uniform mutation (Gao et al., 2012). An improved strategy which combined with Bayesian algorithm was applied for the multilevel thresholding image segmentation optimisation problem (Jiang et al., 2012). The harmony search-based particle swarm optimisation approach was discussed for the application of optimal PID control in electroslag remelting process (Cao and Wang, 2012). A discrete particle swarm optimisation was applied for the optimisation problem of reconfiguration of shipboard power system (Wang et al., 2012). The derivative free particle swarm optimisation was applied for the optimisation problem of dead-time systems control, they introduces two improved forms of PSO algorithm applied to PID controller and Smith predictor design for a class of time delay systems, derivative free optimisation methods, namely simplex derivative pattern search and implicit filtering are used to hybridise PSO algorithm with improved convergence than original PSO (Kanthaswamy and Jovitha, 2011). Simulated annealing-based particle swarm optimisation with adaptive jump strategy was also found the application in the optimisation problem of modelling of dynamic cerebral pressure autoregulation mechanism, in this algorithm, swarm particles jump into the space to find new solutions, the jump radius is selected adaptively based on the particle velocity and its distance from the global best position (Shiru et al., 2011). The bat algorithm was discussed for the application of multi-objective optimisation problem, he proposed a bat-inspired algorithm for solving non-linear, global optimisation problems, and extended this algorithm to solve multi-objective optimisation problems, the algorithm is first validated against a subset of test functions, and then applied to solve multi-objective design problems such as welded beam design (Yang, 2011).

In this article a new PSO technique for MOOP is proposed. Our approach introduces a dynamic inertia weight w in standard PSO to improve algorithm's ability of exploitation and exploration. By using this new strategy, MOPSO algorithm in this paper can find solutions with a good diversity and convergence. The results show that our approach generates satisfactory approximation of the Pareto front with evenly distributed solution along it.

The rest of this paper is organised as follows: the MOOP is formulated in Section 2. The standard PSO model is introduced firstly, and then the approach and implementation details of our method are described in Section 3. Section 4 provides the simulation results and the performance assessment. Section 5 concludes the paper.

2 Description of multi-objective optimisation model

Many real-world problems are inherently of a multi-objective nature with conflicting issues. Generally, multi-objective optimisation problem consists of minimising or maximising the vector function:

$$f(x) = \left[f_1(x), f_1(x), \cdots f_m(x)\right]^T \tag{1}$$

Subject to J inequality and K equality constraints are as follows:

$$\begin{aligned} & g_j(x) \geq 0 \quad j = 1, 2, \cdots J \\ & h_k(x) = 0 \quad k = 1, 2, \cdots K \end{aligned} \tag{2}$$

where $x = [x_1, x_2, x_n]^T \in S$ is the vector of decision variables, and S is the feasible region.

A decision vector $x_1 \in S$ is said to dominate a decision vector $x_2 \in S$ (denoted $x_1 \prec x_2$) iff:

- the decision vector x_1 is not worse than x_2 in all objectives, or $f_i(x_1) \leq f_i(x_2)$, $\forall i = 1, 2, \cdots m$
- the decision vector x_1 is strictly better than x_2 in at least one objective, or $f_i(x_1) < f_i(x_2)$, $i = 1, 2, \cdots m$ for at least one $i = 1, 2, \cdots m$.

And x_1 weakly dominated x_2 (denoted $x_1 \preceq x_2$) iff:

- the decision vector x_1 is not worse than x_2 in all objective, or $f_i(x_1) \leq f_i(x_2)$, $\forall i = 1, 2, \cdots m$

A decision vector $x_1 \in S$ is called Pareto-optimal if there does not exist another $x_2 \in S$ that dominates it. Finally, an objective vector is called Pareto-optimal if corresponding decision vector is Pareto-optimal.

3 Proposed multi-objective optimisation particle swarm algorithm

In this paper, we present a multi-objective particle swarm optimisation algorithm to deal with MOOPs. Firstly, because the inertia weight w is a very important parameter in standard version, it can control algorithm's ability of exploitation and exploration so the accumulation factor of the swarm is introduced in the new algorithm, and the inertia weight is formulated as the function of the factor. In each generation, the w is changed dynamically according to the accumulation factor.

3.1 Particle swarm optimisation algorithm

PSO is a population-based multi-point search technique that mimics the social behaviour of a flock of birds and a fish school, etc. Each single solution in the *n*-dimensional search space is a 'bird' or 'particle', and each solution or particle has a fitness value that is evaluated by a fitness function. The objective of the algorithm is to find the solution that maximise (or minimise) the fitness function. The search starts with a population of M particles called swarm with a random uniform distribution in the *n*-dimensional search space R^N.

Assume the fitness function to be a real function f whose domain is the n-dimensional search space:

$$f : R^N \to R$$

The M particles will be represented by its position x and velocity v in the search space, that is:

$$x_i \in R^N \quad \forall i \in (1, 2, \cdots, M)$$

$$v_i \in R^N \quad \forall i \in (1, 2, \cdots, M)$$

Each particle is encoded by a position vector (initially chosen at random) and the position is updated by using its velocity (initially chosen at random) in successive iterations. The velocity is updated using its own previous best position (*pbest*) and by following the best neighbourhood particle's position (*nbest*). The position and velocity update equations are:

$$v_i^{(k+1)} = wv_i^k + c_1 r_1 \left(Pb_i^k - x_i^k\right) + c_2 r_2 \left(Pb^k - x_i^k\right) \tag{3}$$

$$x_i^{(k+1)} = x_i^k + v_i^{(k+1)} \tag{4}$$

where r_1 and r_2 are real random variable with a normal distribution in the interval (0, 1); c_1 and c_2 are real learning factors, and usually both have the same value; Pb_i^k is the best position found for the i-particle until the k-iteration; Gb^k is the best position for the swarm; w is an scalar factor representing the inertial weight that controls the maximum particle's velocity.

The fitness of a particle is determined by the objective functions. The iterative PSO is performed till some termination criterion is attained, for example, maximum number of iterations or a desired fitness. The performance of PSO depends on information exchange among the particles, which is influenced by neighbourhood topology. The global best (*gbest*) and local best (*lbest*) are two widely used neighbourhood topologies.

3.2 Improve multi-objective optimisation algorithm

In particle swarm optimisation algorithm, the inertia weight influences the trade-off between global and local exploration abilities of the particle. Generally, a larger inertia weight w facilitates global exploration while a smaller inertia weight tends to facilitate local exploration to fine-tune the local search area. Usually a decreasing inertia weight model is used to balance them. In the model, once the swarm finds the local area where the optimal solution resides, the particles would be attracted toward the neighbourhood. The trajectory of each particle is pinned to its best ever visited solution. If better solutions are discovered, each particle updates its current solution for seeking out even better solutions. As such, the search spread is getting tight step by step. Consider a situation that the final destination is only a 'local' optimal solution in that area. When converged, the decreasing inertia weight must turn out to be an extremely small value and the particles hardly escape from the local optimal area. A reasonable strategy for improvement can be envisioned that if a particle finds a better solution then more energy (i.e., weight) is given onto the current velocity to speed up exploitation, and vice versa. Under such circumstances, extended search and/or local refinement can be realised.

To achieve the foregoing concept, this paper proposes a novel non-linear function regulating inertia weight adaptation with a dynamic mechanism for enhancing the performance of PSO algorithms. The non-linear function is given by:

$$w = w_0\left(1-\frac{iter_{current}}{iter_{total}}\right)^r \tag{5}$$

where $iter_{total}$ and $iter_{current}$ represent the total number of iterations and the current iteration number at present time step respectively; $\left(1-\frac{iter_{current}}{iter_{total}}\right) \in (0,1)$ is decreasing with the iteration, which maintains a better convergence degree; $w_0 \in (0, 1)$ is the initial inertia weight, and r measures the uniformly distributed conditions of Pareto optimal solutions in the objective space, it can be denoted as:

$$r = \frac{1}{N*L}\sum_{i=1}^{N}\sqrt{\sum_{d=1}^{n}\left(p_{id}-p_d\right)^2} \in (0,1) \tag{6}$$

where N is the population size, n is the number of variables, L is the length of the maximum diagonal in the search space, p_{id} indicates the d_{th} position of the i_{th} particle, p_d indicates the average values of all particles at the d_{th} position. The smaller the value of s, the more centralised the swarm is. When the swarm is centralised, it becomes difficult for the algorithm to break away from the local optimum.

This mechanism wishes to make particles fly more quickly toward the potential optimal solution, and then through decreasing inertia weight to perform local refinement around the neighbourhood of the optimal solution.

3.3 Steps of improve multi-objective optimisation algorithm

From above, the proposed multi-objective particle swarm optimisation algorithm has the following steps:

Step 1 Initialise the particles

Initialise the population $P(t)$, and given the swarm size N. Initialise an array of particles with random positions and initialise the speed of each particle with 0. Copy non-dominated members of $P(t)$ to P set $t = 1$, for $i = 1$ to N, $pbesti(t) = pi(t)$, $pi(t)$ indicates the ith particle in $P(t)$, and let $vi(t) = 0$; evaluate each of the particles in $P(1)$, and store the position of the particles that represent non-dominated vectors in the repository.

Step 2 Update the particle's position and velocity

Update the positions and velocities of the particles according to equations (3) and (4), which form new population $P'(t + 1)$. Firstly, we compute the crowding-distances of all particles in P, and choose the one with the biggest crowding-distance as $gbest(t)$, which makes it easy to break away from local optimum. Also, by the distances we can compute the inertia weight using equation (5).

Step 3 Evaluate particles

Copy the non-dominated members of $P'(t + 1)$ to P, and remove the dominated members from P. Compute fitness value by fitness function for all particles. When the current position of the particle is better than the position contained in its memory, the particle's position is updated using $pbesti(t) = Pi(t)$. The criterion to decide what position from memory should be retained is simply to apply the Pareto dominance. Set $t = t + 1$.

Step 4 Obtain the optimal solution

Loop to Step 2 until a stopping criterion is met, usually a given maximum generations.

4 Simulation results and analysis

The proposed method is evaluated via computer simulation by choosing three benchmark functions. All experiments were performed in MATLAB 7.0. The parameter is described as follows: swarm size $N = 100$, $r1$ and $r2$ are the random numbers in [0, 1], $c1$ and $c2$ are positive constants. n is the number of the decision variables. Number of generations: 250.

Each of the test functions defined below is structured in the same manner:

$$\begin{aligned}&\min \quad F(x)=\left(f_1(x), f_2(x)\right)\\&\text{s.t.} \quad f_2(x)=g(x)h\left(f_1\left(x_1\right), g(x)\right)\end{aligned} \tag{7}$$

where x = $(x_1, x_2 \cdots x_n)$.

The first function is a convex function, where n = 30, $x_i \in (0, 1)$:

$$\begin{aligned}&ZDT1: f_1\left(x_1\right)=x_1\\&g(x)=1+9\sum_{i=2}^{n} x_i \Big/ (n-1)\\&h\left(f_1, g\right)=1-\sqrt{f_1/g}\end{aligned} \tag{8}$$

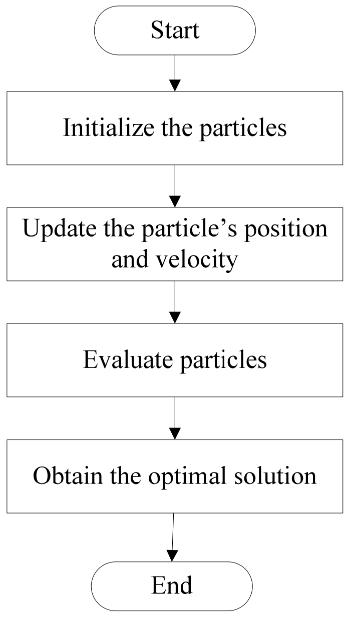
The second function is a non-convex function, where $n = 30, x_i \in (0, 1)$:

$$\begin{aligned}&ZDT2: f_1\left(x_1\right)=x_1\\&g(x)=1+9\sum_{i=2}^{n} x_i \Big/ (n-1)\\&h\left(f_1, g\right)=1-\left(f_1/g\right)^2\end{aligned} \tag{9}$$

The third function is a non-continuous function, where $n = 30, x_i \in (0, 1)$:

$$\begin{aligned}&ZDT3: f_1\left(x_1\right)=x_1\\&g(x)=1+9\sum_{i=2}^{n} x_i \Big/ (n-1)\\&h\left(f_1, g\right)=1-\sqrt{f_1/g}-\left(f_1/g\right)\sin\left(10\pi f_1\right)\end{aligned} \tag{10}$$

Figure 1 Flowchart of improved algorithm


Start
Initialize the particles
Update the particle's position and velocity
Evaluate particles
Obtain the optimal solution
End

As shown in Figures 1 and 2, the curve in each figure is the Pareto-optimal front of each test problem. In our experiment, the proposed method can obtain well distributed Pareto fronts, and the Pareto fronts for each test function are very close to the Pareto-optimal fronts and to some extent, the results are scattered uniformly over the entire Pareto-optimal front.

Figure 2 Convex function test ZDT1

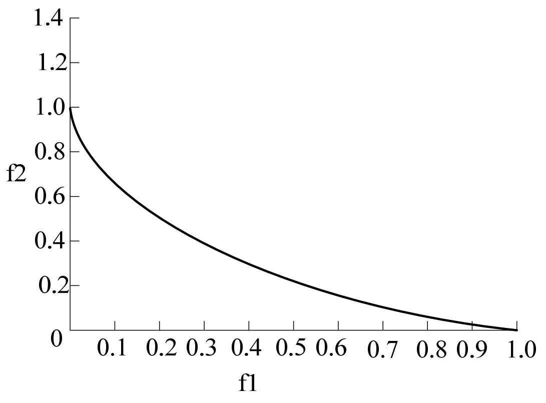


Figure 3 Convex function test ZDT2

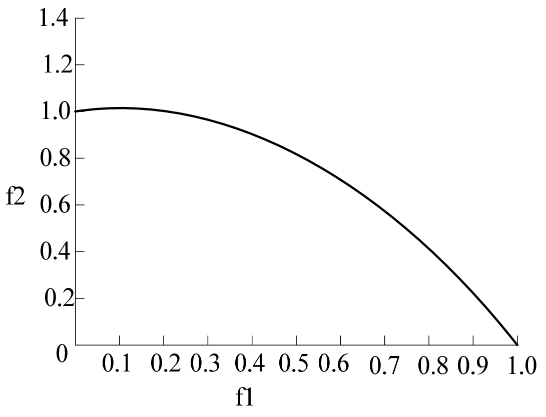
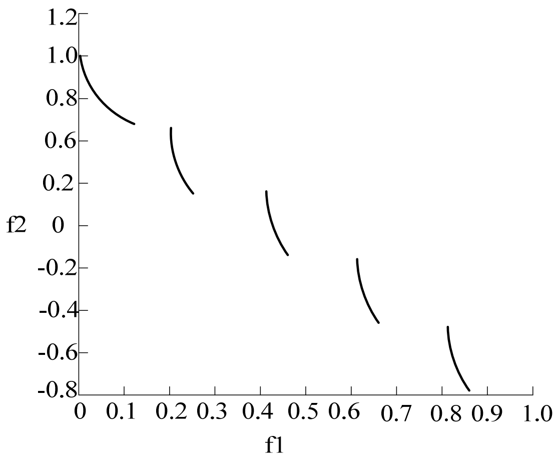


Figure 4 Convex function test ZDT3



We also used two issues are normally taken into consideration, describe as below:

1 minimise the distance of the Pareto front produced by our algorithm with respect to the global Pareto front

2 maximise the spread of solutions found, so that we can have a distribution of vectors as smooth and uniform as possible.

Based on this notion, we adopted one metric to evaluate each of three aspects previously indicated.

1 Generational distance (GD): the concept of generational distance was introduced by Van Veldhuizen and Lamont as a way of estimating how far the elements are in the set of non-dominated vector found so far from those in the Pareto optimal set and is defined as:

$$GD = \sqrt{\sum_{i=1}^{n} d_i^2 \Big/ n} \tag{11}$$

where n is the number of vectors in the set of non-dominated solutions found so far and d_i is the Euclidean distance (measured in objective space) between each of these and the nearest member of the Pareto optimal set.

2 Spacing (SP): Here, one desires to measure the spread of vectors throughout the non-dominated vectors found so far. Schott proposed such a metric measuring the range variance of neighbouring vectors in the non-dominated vectors found so far. This metric is defined as:

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(d - d_i\right)^2} \tag{12}$$

where $d_i = \min_j (| f_1^i(\bar{x}) - f_1^j(\bar{x}) | + | f_2^i(\bar{x}) - f_2^j(\bar{x}) |)$ is the mean of all $di(i, j = 1, 2 \ldots, n)$, and n is the number of non-dominated vectors found so far.

Table 1 shows the comparison of results between NSGA2 and the algorithm in this paper considering the metrics previously described. It can be seen that, the average performance of the presented algorithm is better. In the table, the results of NSGA2 are from Zitzler et al. (2000).

Table1 Results of generational distance and spacing metrics for test function

Algorithm	*ZDT1*		*ZDT2*		*ZDT3*	
	GD	*S*	*GD*	*S*	*GD*	*S*
NSGA2	8.94e–4	0.463	8.24e–4	0.435	4.34e–2	0.576
Presented	9.32e–3	0.702	8.53e–3	0.631	5.97e–2	0.789

5 Conclusions

A new improved PSO for multi-objective problems is described in this paper. As the inertia weight influences the trade-off between global and local exploration abilities of the particle, by taking into account the inertia weight w in standard particle swarm optimisation, a dynamic inertia weight is introduced to improve algorithm's ability of exploitation and exploration. Using this new strategy, multi-objective particle swarm optimisation algorithm in this paper can find solutions with a good diversity and convergence. If a particle finds a better solution then more energy (i.e., weight) is given onto the current velocity to speed up exploitation, and vice versa. Under such circumstances, extended search and local refinement can be realised. The computer simulations for three well-known and difficult benchmark functions taken from the multi-objective optimisation literature are used to evaluate the performance of the proposed approach. The proposed multi-objective optimisation particle swarm method is usually effective to maintain the population diversity and hold a fast convergence velocity, and the new algorithm is able to find uniformly distributed Pareto optimal solutions and is able to converge to the Pareto-optimal front.

Acknowledgements

This work is supported in part by the Scientific Research Foundation of Nanjing Institute of Technology (Grant No. YKJ201110). We are grateful to express our thanks. We are also grateful for the computing resources provided by colleagues in Department of Computer Engineering. The authors also want to give their thanks to colleagues in Nanjing Institute of Technology and Nanjing University of Technology.

References

Cagnina, L. and Esquivel, S. (2005) 'A particle swarm optimizer for multi-objective optimization', *JCS and T*, Vol. 5, No. 4, pp.204–210.

Cao, F. and Wang, W. (2012) 'Harmony search based particle swarm optimisation approach for optimal PID control in electroslag remelting process', *International Journal of Modelling, Identification and Control*, Vol. 15, No. 1, pp.20–27.

Coello, C.A. and Lechuga, M.S. (2002) 'A proposal for multiple objective particle swarm optimizations', in *Proc. Congress on Evolutionary Computation (CEC)*, pp.1051–1056.

Ganguly, S., Sahoo, N.C. and Das, D. (2010) 'A novel multi-objective PSO for electrical distribution system planning incorporating distributed generation', *Energy System*, Vol. 5, No. 2, pp.913–960.

Gao, Y.L., Chen, Y.Z. and Jiang, Q.Y. (2012) 'Multi-objective differential evolution algorithm based on the non-uniform mutation', *International Journal of Modelling, Identification and Control*, Vol. 15, No. 4, pp.284–289.

Hu, X. and Eberhart, R. (2002) 'Multi-objective optimization using dynamic neighborhood particle swarm optimization', in *Proc. IEEE World Congress on Computational Intelligence*, Piscataway, IEEE Service Center, NJ, pp.1677–1681.

Jiang, Y.Z., Hao, Z.F., Yuan, G.Z. and Yang, Z.L. (2012) 'Multilevel thre-sholding for image segmentation through Bayesian particle swarm optimisation', *International Journal of Modelling, Identification and Control*, Vol. 15, No. 4, pp.267–276.

Kanthaswamy, G. and Jovitha, J. (2011) 'Control of dead-time systems using derivative free particle swarm optimisation', *International Journal of Bio-Inspired Computation*, Vol. 3, No. 2, pp.85–102.

Kennedy, J. and Eberhart, R.C.(1995) 'Particle swarm optimization', in *Proc. of IEEE International Conference on Neural Networks*, Pearth, Australia, pp.1942–1948.

Mostaghim, S. and Teich, J. (2003) 'Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO)', in *Proc. IEEE Swarm Intelligence Symposium*, Indiana, USA, pp.26–33.

Mostaghim, S. and Teich, J. (2004) *Covering Pareto-optimal Fronts by Sub Swarms in Multi-Objective Particle Swarm Optimization*, IEEE Press, New York, pp.1404–1410.

Parsopoulos, K.E. and Vrahatis, M.M. (2002) 'Particle swarm optimization method in multi-objective problems', in *Proc. Symposium on Applied Computing*, pp.603–607.

Shinn, J.H. and Shinn, Y.H. (2006) 'Intelligent particle swarm optimization in multi-objective problems', *Advances in Knowledge Discovery and Data Mining*, Vol. 39, No. 18, pp.790–800.

Shiru, S., Ranjana, P., Neeraj, S. and Tiwari, J.P. (2011) 'Simulated annealing-based particle swarm optimisation with adaptive jump strategy for modelling of dynamic cerebral pressure autoregulation mechanism', *International Journal of Bio-Inspired Computation*, Vol. 3, No. 4, pp.225–237.

Wang, L., Liu, Y.H. and Xu, Y.O. (2006) *Multi-objective PSO Algorithm Based on Fitness Sharing and Online Elite Archiving*, IEEE Press, USA, pp.87–94.

Wang, Z., Zhao, D.J., Wang, Y.J. and Liu, D.B. (2012) 'Reconfiguration of shipboard power system using discrete particle swarm optimisation', *International Journal of Modelling, Identification and Control*, Vol. 15, No. 4, pp.277–283.

Wei, J.X. and Wang, Y.P. (2007) *A New Model Based Multi-objective PSO Algorithm*, IEEE Press, USA, pp.87–94.

Wickramasinghe, W.R. and Li, X. (2007) *Choosing Leaders for Multi-objective PSO Algorithm Using Differential Evolution*, IEEE Press, USA, pp.249–258.

Yang, X.S. (2011) 'Bat algorithm for multi-objective optimisation', *International Journal of Bio-Inspired Computation*, Vol. 3, No. 5, pp.267–274.

Zhang, J.D., Lu, J.G., Li, H.L. and Xie, M. (2011) 'Hybrid particle swarm optimisation algorithm for image segmentation', *International Journal of Modelling, Identification and Control*, Vol. 14, No. 4, pp.317–323.

Zitzler, E., Deb, K. and Thiele, L. (2000) 'Comparison of multi-objective evolutionary algorithms: empirical results', *Evolutionary Computation*, Vol. 8, No. 2, pp.1–24.