VOL. 10, NO. 23, DECEMBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

MLPSO: MULTI-LEADER PARTICLE SWARM OPTIMIZATION FOR MULTI-OBJECTIVE OPTIMIZATION PROBLEMS

Zuwairie Ibrahim[1], Kian Sheng Lim[2], Salinda Buyamin[2], Siti Nurzulaikha Satiman[1], Mohd Helmi Suib[1], Badaruddin Muhammad[1], Mohd Riduwan Ghazali[1], Mohd Saberi Mohamad[3], and Junzo Watada[4]
[1]Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang, Pekan, Malaysia
[2]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia
[3]Faculty of Computing, Universiti Teknologi Malaysia, Johor, Malaysia
[4]Graduate School of Information, Production, and Systems, Waseda University, Japan
E-Mail: zuwairie@ump.edu.my

ABSTRACT

The particle swarm optimization (PSO) algorithm, which uses the best experience of an individual and its neighborhood to find the optimum solution, has proven useful in solving various optimization problems, including multi-objective optimization (MOO) problems. In MOO problems, existing multi-objective PSO algorithms use one or two leaders to guide the movement of every particle in a search space. This study introduces the concept of multiple leaders to guide the particles in solving MOO problems. In the proposed Multi-Leader PSO (MLPSO) algorithm, the movement of a particle is determined by all leaders that dominate that particle. This concept allows for more information sharing between particles. The performance of the MLPSO is assessed by several benchmark test problems, with their convergence and diversity values are computed. Solutions with good convergence and diversity prove the superiority of the proposed algorithm over MOPSOrand algorithm.

Keywords: particle swarm optimization, multi-objective optimization, multiple leaders, convergence, diversity.

INTRODUCTION

Particle Swarm Optimization (PSO) is a population-based stochastic optimization algorithm, inspired by the social behavior of bird flocking [1]. The PSO algorithm, introduced by Kennedy and Eberhart, consists of a group of particles that traverse a search space seeking an optimum solution according to a particular objective function. The movement of a particle is subjected to its own best found solution and the best found solution in the neighborhood known as leader.

Multi-objective Optimization (MOO) is a type of optimization problem that involves more than one objective function. These objective functions often conflict with each other, and hence, it is not possible to find a single best solution. Thus, a variant of the PSO algorithms known as Multi-Objective PSO (MOPSO) has been introduced for solving MOO problems [2]. Usually, an external archive is used to store a set of non-dominated solutions and the leader is determined based on the non-dominate solutions.

A variety of Multi-objective PSO algorithms have been published to date. From the literature review, it is found that most MOPSO algorithms use a leader to guide particles' movement but with different selection strategy. For example, the leader can be selected from the non-dominated solutions based on their fitness value, such as crowding distance [3], particle influence on diversity to the Pareto front [4], niche count [5], the nearest neighbor density estimator [5], and the sigma value [6].

Besides, the leader can be selected from the non-dominated solutions according to their location in objective space. A number of approaches have been investigated. As such, non-dominated solutions can be divided into several hypercubes [7] before the selection. In [8], the dominate trees data structure [9] is used to select leader from the nearest members in objective space. The "stripes" mechanism [10] divides the objective space into several stripes for leader selection. The leader can also be determined according to the domination between a particle and the non-dominated solutions [11].

More than one leader approach has been introduced in [12]. In particular, two different "guiders", or leader, has been employed in solving the MOO problems. The first guider is randomly chosen from the non-dominated solutions, while the second guider is the non-dominated solution with the largest crowding distance value.

Since the search in MOPSO algorithms is driven by an external archive, a number of non-dominated solutions in the archive may benefit different particles. In other words, each particle may need a different set of non-dominated solutions as leader. This mechanism would prevent a particular particle from being trapped in local optima, which is the natural weakness of PSO algorithm [13]. In this study, an alternative algorithm, which is based on more than two leaders, is proposed. The proposed algorithm is called Multi-Leader PSO (MLPSO).

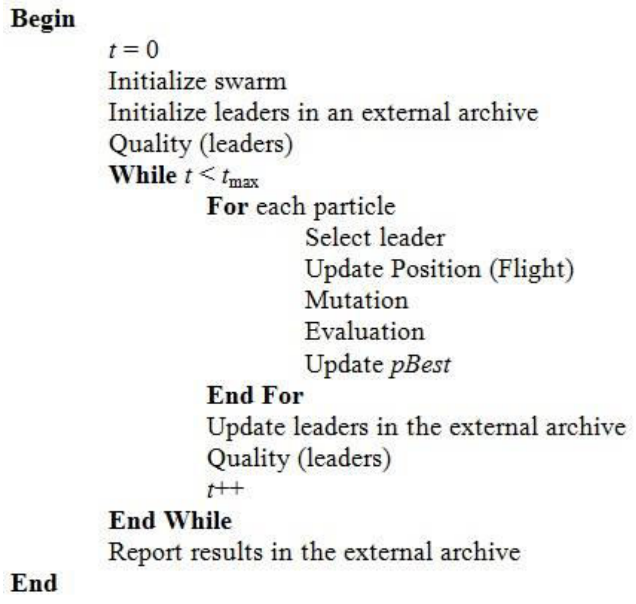
Note that similar concept has been employed in [14]. However, different equation has been used in updating the position of particles. Moreover, they solved multi-objective problems in binary search space.

PARTICLE SWARM OPTIMIZATION

Consider an N-dimensional search space and I number of particles. The fitness is calculated based on each particle's position, $\boldsymbol{p}_i$ (i = 1, 2, …, I). The velocity, $\boldsymbol{v}_i(t)$, is updated for each dimension with the cooperation of the best position found by the particle, $\boldsymbol{pBest_i}$, and among all particles, $\boldsymbol{gBest}$, as in Equation (1).

$$\boldsymbol{v}_i(t+1) = \omega\boldsymbol{v}_i(t) + c_1 r_1[\boldsymbol{pBest}_i - \boldsymbol{p}_i(t)] + c_2 r_2[\boldsymbol{gBest} - \boldsymbol{p}_i(t)] \tag{1}$$

where ω is the inertia weight. The values c_1 and c_2 are the cognitive and social coefficients, respectively, and both are positive constants that define the tendency to move toward ***pBest*** and ***gBest***, while r_1 and r_2 are both standard uniformly distributed random values, $U[0,1]$. For each dimension, the particle position is then updated according to Equation (2).



```
Begin
        t = 0
        Initialize swarm
        Initialize leaders in an external archive
        Quality (leaders)
        While $t < t_{max}$
                For each particle
                        Select leader
                        Update Position (Flight)
                        Mutation
                        Evaluation
                        Update pBest
                End For
                Update leaders in the external archive
                Quality (leaders)
                t++
        End While
        Report results in the external archive
End
```

Figure-1. Pseudocode for general MOPSO algorithm.

$$\boldsymbol{p}_i(t+1) = \boldsymbol{p}_i(t) + \boldsymbol{v}_i(t+1) \tag{2}$$

MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZATION

Figure-1 illustrates the pseudocode for the general Multi Objective PSO algorithm [15]. At iteration $t = 0$, the algorithm begins by initializing all particles, associating their positions and velocities with random values. Then, the fitness of each particle is calculated based on the particle's position. Based on the fitness values, a set of non-dominated particles is stored in an external archive. The non-dominated particles are subjected to a quality measure to determine the quality of each non-dominated particle. Then, each particle selects a non-dominated solution from the archive as leader to update its velocity and position based on Equation (1) and Equation (2), respectively. Then, a mutation operation is performed before the ***pBest*** is updated, based on Pareto dominance. Once all particles are updated, the archive will be updated with the latest non-dominated solutions. At the end of the iteration, the solutions in the archive are reported as the final optimum solutions for the MOO problem. Note that in Figure-1, $t_{\max}$ indicates the maximum number of iterations.

THE PROPOSED MULTI-LEADER PSO (MLPSO)

The MLPSO algorithm includes two equal-size swarms where both swarms employ the RANDOM selection method [11] to determine the leaders. The difference is that the particles in the first swarm require the multiple leaders to fully utilise the information from all leaders. While, the second swarm operates like regular MOO algorithm by optimised using the information from one leader only. Figure-2 shows the flow chart of the proposed MLPSO algorithm. The solid line and the dotted line represent the flow of the first and the second swarms, respectively. Also in Figure-2, $t_{\max}$ indicates the maximum number of iterations.

During initialization, all particle positions in both swarms are initialised with a uniformly distributed random value within the search space limit for all N dimensions, as shown in Equation (3).

$$p_{s,i}\,(t=0) = \{\, p_{s,i,n},\ n = 1, 2, \ldots, N\} = U[\boldsymbol{p}^l, \boldsymbol{p}^u] \quad \text{for } i = 1, 2, \ldots, I \text{ and } s = 1, 2 \tag{3}$$

where N is the number of dimensions of the search space, s is the index of the swarm, I is the size of a particle in the swarm, and the uniform distribution, $U[\boldsymbol{p}^l, \boldsymbol{p}^u]$, is bounded between the lower limit, $\boldsymbol{p}^l$, and the upper limit, $\boldsymbol{p}^u$, of the N dimensional search space. The velocity and $\boldsymbol{pBest_i}$ are initialised to zero and particle position, respectively.

Then, the particle position is used to calculate all M objective functions. During the computation, $\boldsymbol{pBest_i}$ is replaced with the new position if the previous $\boldsymbol{pBest_i}$ is dominated by the new position or if both the previous $\boldsymbol{pBest_i}$ and the new position are non-dominated by each other.

An external archive, Ψ, is important in MOPSO algorithm to store non-dominated solutions, $\boldsymbol{\psi}$. During computation, the external archive is updated to store the latest found non-dominated solutions. In this work, the newly found non-dominated solutions of both swarms are involved in the archive update. Based on the *Pareto Optimality* concept, if a new solution is not dominated by all solutions in the archive, then the solution is added into the archive. Whenever any solution in the archive is dominated by the new solution, it is deleted from the archive.

During computation, the archive size, α, can easily reach its limit, $\alpha_{\max}$. Thus, another mechanism is required to remove a solution from the external archive. In this work, the solution with the smallest crowding distance [16] value is removed to ensure that the non- dominated solutions are not too crowded at a certain area in the Pareto front.

The RANDOM method [11] was adopted as the leader selection mechanism, with some changes to allow for the selection of a set of multiple leaders.

Definition : The multiple leader set, $ML_{s,i}$, is defined as the set of non-dominated solutions that dominate the i-th particle from the s-th swarm as below:

$$ML_{s,i} = \{ \psi \prec \boldsymbol{p}_{s,i} \mid \psi \in \Psi \} \tag{4}$$

Consider the particles and the non-dominated solutions shown in Figure-3. For example, the first particle labelled with '1' is dominated by the non-dominated solution 'A', and thus, $ML_{s,1} = \{\boldsymbol{\psi}_A\}$. The second particle is dominated by the non-dominated solutions 'A' and 'B', and thus, $ML_{s,2} = \{\boldsymbol{\psi}_A, \boldsymbol{\psi}_B\}$. Similarly, $ML_{s,3} = \{\boldsymbol{\psi}_A, \boldsymbol{\psi}_B, \boldsymbol{\psi}_C\}$ because the non-dominated solutions 'A', 'B', and 'C' dominate the third particle. For the rest of the particles, $ML_{s,4} = \{\boldsymbol{\psi}_B, \boldsymbol{\psi}_C\}$ and $ML_{s,5} = \{\boldsymbol{\psi}_C\}$. Furthermore, if the particle is a member of the archive, then the other members in the archive are selected to be its leaders. In this case, $ML_{s,A} = \{\boldsymbol{\psi}_B, \boldsymbol{\psi}_C\}$.

In MLPSO, the non-dominated solutions $\boldsymbol{\psi}$ in the $ML_{s,i}$ are the leaders (or ***gBest***) which dominate the $\boldsymbol{p}_{s,i}$. The $ML_{s,i}$ can be represented as follows:

$$ML_{s,i} = \{ \boldsymbol{gBest}_{s,i,1}, \boldsymbol{gBest}_{s,i,2}, \ldots, \boldsymbol{gBest}_{s,i,q}, \ldots \boldsymbol{gBest}_{s,i,|ML_{s,i}|} \} \tag{5}$$

where $|ML_{s,i}|$ represents the cardinality of $ML_{s,i}$ or the number of leaders in the leader set, and $\boldsymbol{gBest}_{s,i,q}$ is the q-th leader in the $ML_{s,i}$.

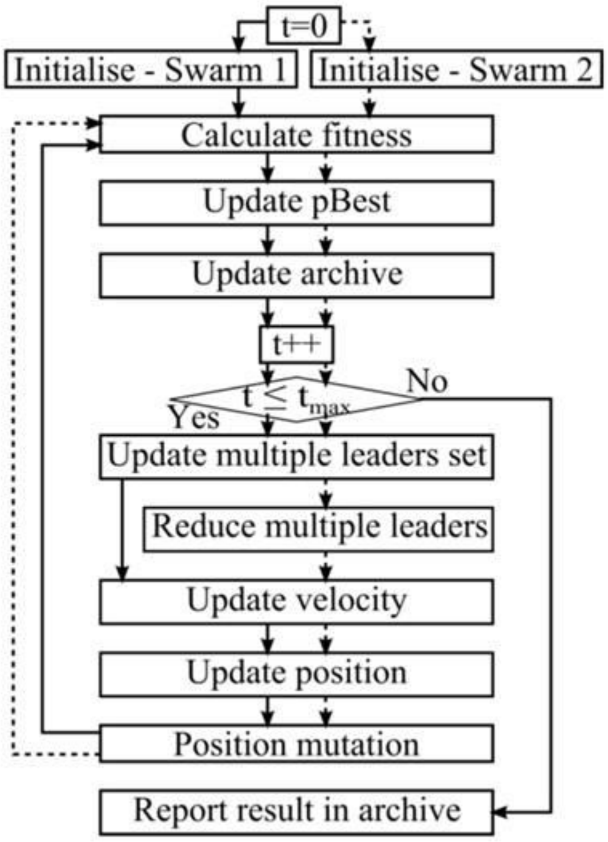


Figure-2. The MLPSO algorithm.

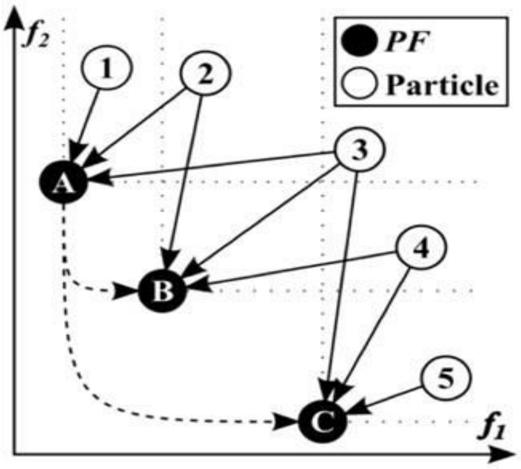


Figure-3. Particles and non-dominated solutions at the Pareto Front (PF).

In addition, the particles in the first swarm used all leaders while the second swarm require only one leader for the guidance in moving the particles for respective swarm. The single leader can be obtained by reducing the $ML_{2,i}$ size randomly with equal probability.

The modified velocity update based on the new concept of multiple leaders is shown in Equation (6).

$$\boldsymbol{v}_{s,i}(t+1) = \omega \boldsymbol{v}_{s,i}(t) + c_1 r_1 [\boldsymbol{pBest}_n^{s,1} - \boldsymbol{p}_{s,i}(t)] + c_2 r_2 \sum_{q=1}^{|ML_{s,i}|} c_q [\boldsymbol{gBest}_{s,i,q} - \boldsymbol{p}_{s,i}(t)] \tag{6}$$

Specifically, the social term is replaced with $ML_{s,i}$.

However, it is possible that $v_{s,i}$ $(t + 1)$ value become very large, thus, a weight gain, c_q, is multiplied in the social term. Besides, it is desirable to explore the region with less non-dominated solutions in Pareto front. Hence, c_q should vary inversely proportional to the density of solutions around a leader. Thus, both swarms use the same velocity equation with c_q is calculated based on Equation (6) as to promote diversity performance.

$$c_q = \frac{1}{1 + |ML_{s,i}| e^{-|ML_{s,i}| CD_q}} \tag{7}$$

where CD_q is the crowding distance value for the q-th leader, evaluated based on the (q+1)-th and (q-1)-th non-dominated solutions that are sort ascending with respect to the first objective function. The range of the objective functions do not affect the CD_q as it is normalised to the non-dominated solutions that have extreme value at either one of the objective functions. However, the crowding distance for the extreme non-dominated solutions is set to infinity as they only have one neighbour non-dominated solution. After the velocity update, the position of i-th particle in the n-th dimension of s-th swarm is updated conventionally. Both velocity and position updates are clamped at certain limit based on the experiment setup. However, if the position update is clamped, the direction of the velocity updates is inverted.

Some particles are subjected to a position mutation. Polynomial mutation [17], which has been used in NSGA-II [16] and is shown in Equation (8), is applied to a particular dimension with $1/N$ probability.

$$\boldsymbol{p}_{s,i} = \boldsymbol{p}_{s,i} + \Delta\boldsymbol{q}(\boldsymbol{p}_l - \boldsymbol{p}_u),\ \boldsymbol{p}_l \le \boldsymbol{p}_{s,i} \le \boldsymbol{p}_u \tag{8}$$

where N is the number of dimensions for the corresponding problem, and $\boldsymbol{p}_{s,i}$ is the position to be mutated, which is bound between the upper ($\boldsymbol{p}_u$) and lower ($\mathbf{p}_l$) search space limits for each dimension. The polynomial equation is scaled to the upper and lower limits [18], as follows:

$$\Delta q = \begin{cases} \left[2r + (1-2r)\left(1 - \frac{p_{s,i} - p_l}{p_u - p_l}\right)^{z+1}\right]^{1/z+1} - 1 & , r \le 0.5 \\ 1 - \left[2(1-r) + 2(r-0.5)\left(1 - \frac{p_u - p_{s,i}}{p_u - p_l}\right)^{z+1}\right]^{1/z+1} & , otherwise \end{cases} \tag{9}$$

where $r = U[0,1]$, and z is the distribution index for real-coded mutation.

PERFORMANCE MEASURE AND TEST PROBLEM

Four quantitative performance measures have been used to evaluate the performance of the proposed MLPSO: (1) Number of Solutions (*NS*) (2) Generational Distance (*GD*) (3) *Spread* (4) Hypervolume (*HV*).

In this study, the ZDT [19] benchmark test problems are used to validate the performance of the algorithm. These benchmark test problems include six test problems. However, the ZDT5, which is used for binary evaluation, has been excluded because this study focuses on the continuous search space problem. The parameters used for the test problems are based on [19]. Other than ZDT test problems, nine test problems introduced by Walking Fish Group (WFG) [20] were also included to validate the performance of the algorithm.

Table-1. Results based on ZDT test problems.

Problem	Algorithm	NS		GD		Spread		HV	
ZDT1	MOPSOrand	62.180000 (44.574618)	+	0.103789 (0.155258)	+	0.715336 (0.306594)	+	0.248975 (0.253490)	+
	MLPSO	99.370000 (5.900959)		0.004703 (0.008875)		0.406333 (0.324481)		0.495925 (0.233864)	
ZDT2	MOPSOrand	2.590000 (1.189983)	+	0.938974 (0.316596)	+	1.000000 (0.000000)	+	0.000000 (0.000000)	-
	MLPSO	1.310000 (0.917782)		0.120607 (0.167688)		0.999987 (0.000108)		0.000000 (0.000000)	
ZDT3	MOPSOrand	86.110000 (31.869651)	+	0.024158 (0.056863)	+	0.924418 (0.038986)	+	0.059409 (0.061945)	+
	MLPSO	100.0000 (0.000000)		0.003104 (0.004121)		1.081121 (0.075004)		0.313265 (0.114657)	
ZDT4	MOPSOrand	41.470000 (48.038380)	+	7.145317 (6.203850)	+	0.989162 (0.024468)	+	0.000000 (0.000000)	+
	MLPSO	87.540000 (28.885850)		0.201061 (0.474677)		0.642747 (0.328406)		0.380340 (0.279713)	
ZDT6	MOPSOrand	55.790000 (41.736377)	+	0.159572 (0.213036)	+	0.861688 (0.378240)	+	0.142160 (0.279713)	+
	MLPSO	100.000000 (0.000000)		0.002094 (0.002528)		0.238675 (0.196768)		0.400521 (0.000170)	

EXPERIMENTS

In this work, each experiment was repeated for 100 runs to provide reliable and statistical significant results. For fair comparison, each algorithm compared in this work is subjected to 25,000 function evaluations. In each comparison, the average and standard deviation (shown in bracket) of the performance measures are showed. In addition, for each function, Kruskal-Wallis test is performed to determine if there is any difference in performance under each performance measure [21]. All statistical tests are performed with confidence level of 95%. The symbol '+' indicates if there is significant difference in performance and the symbol '-' indicates opposite condition. The best and second best performances are highlighted in dark and light grey, respectively.

The Multi Objective PSO algorithm with RANDOM method (denoted as MOPSOrand) [11], is compared to the proposed algorithm since the MOPSOrand is very similar to the MLPSO algorithm except that the concept of multi leaders is not included. For a fair comparison, both MOPSOrand and MLPSO algorithm used the same parameter setting as in [11]. Both algorithms contain 100 particles and iterate for a maximum of 250 iterations. The $\omega = 0.5$, $c_1 = c_2 = 1.0$, and the archive is limited to 100 solutions only. Besides, none of the particles in both algorithms are subjected to mutation. The MLPSO particles are equally divided into two swarms, according to the proposed MLPSO algorithm.

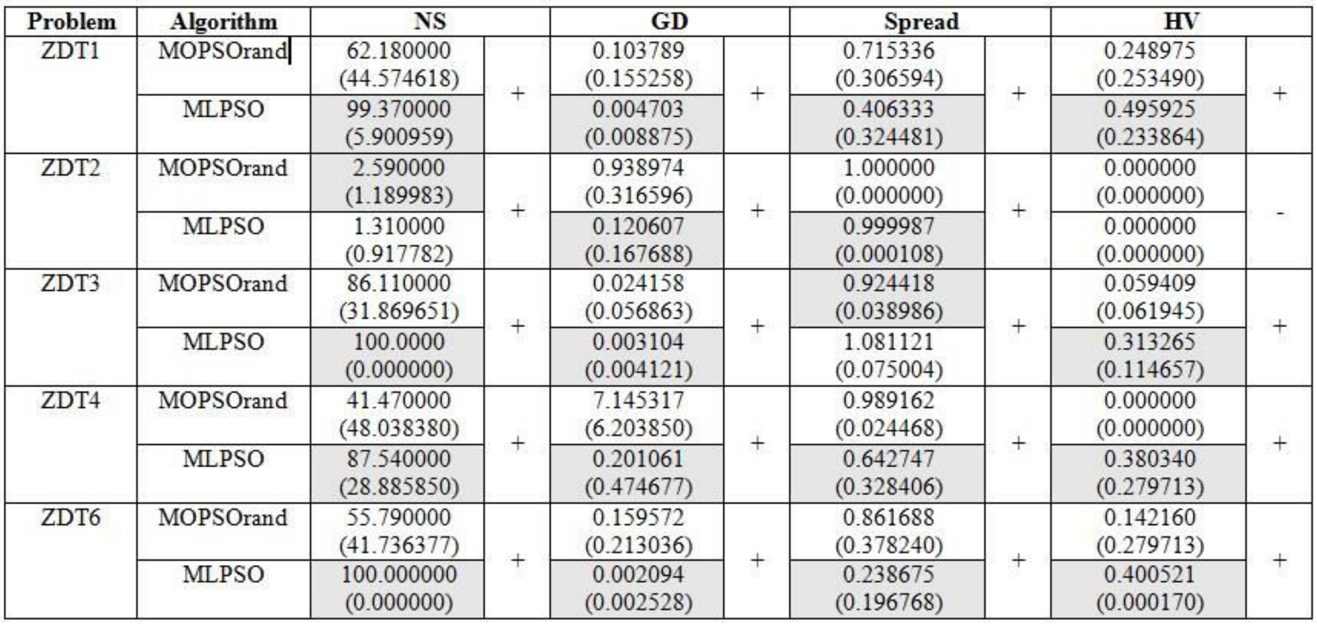
RESULT AND DISCUSSION

The result of performance measures evaluated with MOPSOrand and MLPSO algorithms on ZDT test problems are listed in Table-1. It is observed that the MLPSO outperformed the MOPSOrand in all test problems with significant difference in performance. Besides, the differences of performance measured between both algorithms are relatively large. However, the MOPSOrand only show better result in *NS* and *Spread* measured on ZDT2 and ZDT3 problem, respectively. Even with higher number of solutions in ZDT2 problem, the MOPSOrand was unable to converge. The MLPSO has larger *Spread* value for ZDT3 problem, is negligible when it has superiority in the *GD* and *HV* measures.

Table-2 illustrates the performance measures evaluated on WFG test problems. There is significant difference in performance for most of the performance measure except *NS* as both algorithms obtain similar number of non-dominated solutions. However, in most cases, the MLPSO shows better performance than the MOPSOrand except WFG4 problem. In WFG4 problem, without position mutation, the particles in MLPSO which follow the multi leader set have higher tendency to follow the same group of leader due to the multi-modal feature. However, the differences in performance for both algorithms are quite small in WFG4 problem.

Based on these two set of test problems, MLPSO shows better performance compared to MOPSOrand. Therefore, it can be concluded that the concept of using multiple leaders to guide the particles' movement is practical and proven to obtain a good quality of Pareto front.

Table-2. Results based on WFG test problems.



Problem	Algorithm	NS		GD		Spread		HV	
ZDT1	MOPSOrand	62.180000 (44.574618)	+	0.103789 (0.155258)	+	0.715336 (0.306594)	+	0.248975 (0.253490)	+
	MLPSO	99.370000 (5.900959)		0.004703 (0.008875)		0.406333 (0.324481)		0.495925 (0.233864)	
ZDT2	MOPSOrand	2.590000 (1.189983)	+	0.938974 (0.316596)	+	1.000000 (0.000000)	+	0.000000 (0.000000)	-
	MLPSO	1.310000 (0.917782)		0.120607 (0.167688)		0.999987 (0.000108)		0.000000 (0.000000)	
ZDT3	MOPSOrand	86.110000 (31.869651)	+	0.024158 (0.056863)	+	0.924418 (0.038986)	+	0.059409 (0.061945)	+
	MLPSO	100.0000 (0.000000)		0.003104 (0.004121)		1.081121 (0.075004)		0.313265 (0.114657)	
ZDT4	MOPSOrand	41.470000 (48.038380)	+	7.145317 (6.203850)	+	0.989162 (0.024468)	+	0.000000 (0.000000)	+
	MLPSO	87.540000 (28.885850)		0.201061 (0.474677)		0.642747 (0.328406)		0.380340 (0.279713)	
ZDT6	MOPSOrand	55.790000 (41.736377)	+	0.159572 (0.213036)	+	0.861688 (0.378240)	+	0.142160 (0.279713)	+
	MLPSO	100.000000 (0.000000)		0.002094 (0.002528)		0.238675 (0.196768)		0.400521 (0.000170)	

CONCLUSIONS

The existing Multi Objective PSO algorithms used at most two leaders to update the particle velocity. In this paper, a new implementation of the PSO algorithm for MOO problems based on the guidance of multiple leader guidance is presented. The proposed MLPSO algorithm has been evaluated on several ZDT and WFG test problems to measure the resulting convergence and diversity using four performance measures: *Number of Solution*, *Generational Distance*, *Spread*, and *Hypervolume*. The proposed algorithm showed excellence performance for most cases over the algorithm that it originates from which share similar leader selection mechanism except the concept of multi leaders.

ACKNOWLEDGEMENTS

This work is financially supported by the Fundamental Research Grant Scheme (RDU130117) awarded by the Ministry of Higher Education (MOHE) to Universiti Malaysia Pahang.

REFERENCES

[1] M. Abido. 2010. Multiobjective particle swarm optimization with nondominated local and global sets. Natural Computing, Vol. 9, No. 3, pp. 747-766.

[2] J. Moore and R. Chapman. 1999. Application of Particle Swarm to Multiobjective Optimization. Department of Computer Science and Software Engineering, Auburn University.

[3] T. Ray and K. M. Liew. 2010. A swarm metaphor for multiobjective design optimization. Engineering Optimization, Vol. 34, No. 2, pp. 141 - 153.

[4] T. Bartz-Beielstein, P. Limbourg, J. Mehnen, K. Schmitt, K. E. Parsopoulos, and M. N. Vrahatis. 2003. Particle swarm optimizers for Pareto optimization with enhanced archiving techniques.

VOL. 10, NO. 23, DECEMBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

Congress on Evolutionary Computation (CEC 2003), Vol. 3, pp. 1780-1787.

[5] X. Li. 2003. A non-dominated sorting particle swarm optimizer for multiobjective optimization. Genetic and Evolutionary Computation, pp. 198-198.

[6] S. Mostaghim and J. Teich. 2003. The role of ε-dominance in multi objective particle swarm optimization methods. Congress on Evolutionary Computation (CEC 2003), Vol. 3, pp. 1764-1771.

[7] C. A. Coello Coello and M. S. Lechuga. 2002. MOPSO: a proposal for multiple objective particle swarm optimization. Congress on Evolutionary Computation (CEC 2002), Vol. 2, pp. 1051- 1056.

[8] M. Greeff and A. Engelbrecht. 2010. Dynamic multi-objective optimisation using PSO. Multi-Objective Swarm Intelligent Systems, pp. 105-123.

[9] R. Everson, J. E. Fieldsend, and S. Singh. 2002. Full elite sets for multi-objective optimisation. Adaptive Computing in Design and Manufacture V, pp. 343-354.

[10] M. A. Villalobos-Arias, G. T. Pulido, and C. A. C. Coello. 2005. A proposal to use stripes to maintain diversity in a multi-objective particle swarm optimizer. IEEE Proceedings of the Swarm Intelligence Symposium (SIS2005), pp. 22-29.

[11] J. E. Alvarez-Benitez, R. M. Everson, and J. E. Fieldsend. 2005. A MOPSO algorithm based exclusively on pareto dominance concepts. Evolutionary Multi-Criterion Optimization, pp. 459-473.

[12] M.-T. Pham, D. Zhang, and C. S. Koh. 2012. Multi-guider and cross-searching approach in multi-objective particle swarm optimization for electromagnetic problems. IEEE Transactions on Magnetics, Vol. 48, No. 2, pp. 539-542.

[13] I. Schoeman and A. Engelbrecht. 2005. A parallel vector-based particle swarm pptimizer. Adaptive and Natural Computing Algorithms, pp. 268-271.

[14] M. Shokrian and K. A. High. 2014. Application of a multi objective multi-leader particle swarm optimization algorithm on NLP and MINLP problems. Computers and Chemical Engineering, Vol. 60, pp. 57-75.

[15] M. Reyes-Sierra and C. A. C. Coello. 2006. Multi-objective particle swarm optimizers: a survey of the state-of-the-art. International Journal of Computational Intelligence Research, Vol. 2, No. 3, pp. 287–308.

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 2, pp. 182-197.

[17] K. Deb and M. Goyal. 1996. A combined genetic adaptive search (GeneAS) for engineering design. Computer Science and Informatics, Vol. 26, pp. 30-45.

[18] K. Deb and R. B. Agrawal. 1995. Simulated binary crossover for continuous search space. Complex Systems, Vol. 9, No. 2, pp. 115 - 148.

[19] E. Zitzler, K. Deb, and L. Thiele. 2000. Comparison of multiobjective evolutionary algorithms: empirical results. Evolutionary Computation, Vol. 8, No. 2, pp. 173-195.

[20] S. Huband, L. Barone, L. While, and P. Hingston. 2005. A scalable multi-objective test problem toolkit. Evolutionary multi-criterion optimization, pp. 280-295.

[21] M. Helbig and A. Engelbrecht. 2013. Dynamic multi-objective optimization using PSO. Metaheuristics for Dynamic Optimization, pp. 147-188.