# Multi-Objective Particle Swarm Optimization Algorithms – A Leader Selection Overview

Lim Kian Sheng[*], Zuwairie Ibrahim[+], Salinda Buyamin[*], Anita Ahmad[*], Mohd Zaidi Mohd Tumari[+], Mohd Falfazli Mat Jusof[+], Nor Azlina Ab. Aziz[⁎]

[*]Faculty of Electrical Engineering, Universiti Teknologi Malaysia
Johor Darul Takzim, Malaysia
kiansheng860524@yahoo.com, {salinda, anita}@fke.utm.my

[+]Faculty of Electrical and Electronic Engineering, Universiti Malaysia Pahang
Pahang Darul Makmur, Malaysia
{zuwairie, zaidimt, mfalfazli}@ump.edu.my

[⁎]Faculty of Engineering and Technology, Multimedia University
Melaka, Malaysia
azlina.aziz@mmu.edu.my

***Abstract* — Multi Objective Optimization (MOO) problem involves simultaneous minimization or maximization of many objective functions. Various MOO algorithms have been introduced to solve the MOO problem. Traditional gradient-based techniques are one of the methods used to solve MOO problems. However, in the traditional gradient-based technique only one solution is generated. Thus, an alternative approach such as Particle Swarm Optimization (PSO), which able to produce a number of possible solutions are highly desirable. In PSO, particles search the optimum solution under the influence of a better solution known as leader. This leader facilitates cooperation between all particles. However, this strategy to select the leader has to be changed when it is used for MOO problems. This paper presents an overview of the multi-objective PSO algorithms, which emphasize on the leader selection. In addition, the description of PSO and multi-objective optimization problems are also provided.**

***Keywords* — *Particle Swarm Optimization; multi-objective; leader; Pareto; test problems***

## I. Introduction

James Kennedy and Russell C. Eberhart, who are social psychologist and an electrical engineer, respectively, contributed to the introduction of PSO algorithm, which is a population-based stochastic optimization algorithm inspired by the computer simulation based on choreography of bird flocking [1]. In PSO algorithm, the definition for the represented terms are given below:

Particle: A particle represents birds, which encodes possible solutions to an optimization problem based on its position. In PSO algorithm, a group of particles is known as a swarm of particles, *S*.

Position of particle: The position of particle represents a decision variable with *N*-numbers of parameter value for an optimization problem with *N*-dimensional search space.

Velocity of particle: The velocity of particle represents a vector with equal number of dimension as the position of particle.

Fitness of particle: The fitness of particle represents evaluation of an objective function.

Personal best experienced position: The *pBest* is the position, which has the best fitness ever experienced by a particle.

Global best experienced position: The *gBest* is the position, which has the best fitness ever experienced by all particles in a swarm.

With reference to the standard PSO, as shown in Algorithm 1, these particles explore the defined search space to search for solutions, which best satisfy the objective function of the optimised problem. Thus, each particle collaborates with the others during the search process by comparing its current position with the *pBest* and *gBest*. By considering each iteration as a unit step time (*t*), each *i*-th particles update their velocity at *n*-th dimension with the simplified velocity equation as defined in Eq. 1.

$$v_{i,n}(t+1) = v_{i,n}(t) + C_1R_1[pBest_{i,n} - x_{i,n}(t)] + C_2R_2[gBest_n - x_{i,n}(t)] \quad (1)$$

where $C_1$ and $C_2$ are the cognitive and social constant respectively. These constants control the tendency for particle to land violently (set at large value) or swirl slowly (set at small value) towards the *pBest* or *gBest*. Meanwhile, $R_1$ and $R_2$ are random variables between 0.0 and 1.0. These random variables (stochastic) introduce the craziness into PSO algorithm to prevent the particle from quickly settling on unanimous and unchanging direction. If particles settle too quickly, they are unable to find the best solution especially when there are many local best solutions. If particles settle too slowly the global best solution was not passed through by any particles.

**Algorithm 1** PSO algorithm.

```
1:  for each $particle$ do
2:      random $position$ and $velocity$
3:      evaluate $fitness$
4:      $pBest \leftarrow position$
5:  end for
6:  update $gBest$ (Equation 4.3)
7:  while $termination\_criterion$ is false do
8:      for each $particle$ do
9:          update $velocity$ (Equation 4.4)
10:         update $position$ (Equation 4.5)
11:         evaluate $fitness$
12:         update $pBest$ (Equation 4.2)
13:     end for
14:     update $gBest$ (Equation 4.3)
15:     update $termination\_criterion$
16: end while
```

Once the velocity of each particle is updated, each particle is now able to move to new position using Eq. 2.

$$x_{i,n}(t+1) = x_{i,n}(t) + v_{i,n}(t+1) \tag{2}$$

Throughout iterations, these particles will experience various positions, which represent different possible solutions for the problem. Therefore, the *gBest* at the final iterations will be the best solution for the problem.

## II. Multi-objective Optimization Problems

MOO problems involve simultaneous minimisation or maximisation of multiple objective functions, which are usually conflicting with each other. In general, this work considers only the minimisation problems. Here, some background knowledge related to MOO problems, performance measures used to evaluate the quality of the solutions, and the MOO test problems, are introduced.

### A. Decision Variable

**Definition 1.** The decision variable is a vector of $N$-number of parameter values, which represents a solution for an optimization problem with $N$-dimensional search space.

The parameter values can be either in discrete value, Z, for problem with discrete space or continuous value, R, for problem with continuous search space. These parameter values are denoted as $x_i \in$ Z or $x_i \in$ R for $i = 1,2,...,N$. Thus, the decision variable which represents a solution in an optimization problem is denoted in Eq. 3.

$$\vec{x} = \{x_1, x_2, \ldots, x_N\} \tag{3}$$

### B. Constraint Function

**Definition 2.** Constraint function is the function with respect to the solution such that it can draw a region which contains only the feasible solutions, F.

These constraints can be defined by inequality and equality constraint functions as shown in Eq. (4) and Eq. (5), respectively.

$$g_k(\vec{x}) \leq 0, k = 1,2,\ldots,p \tag{4}$$

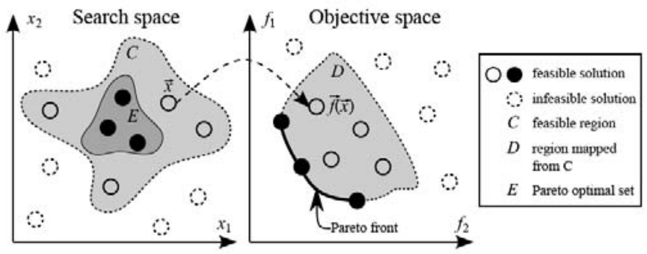


Fig. 1 llustration of search space and objective space.

$$h_l(\vec{x}) = 0, l = 1,2,\ldots,q \tag{5}$$

where $p$ and $q$ are the numbers of inequality and equality constraints respectively. A problem could have more than one constraint functions that have to be satisfied. Besides, there are various kinds of constraint functions which are problem dependent.

### C. Objective Function

**Definition 3**. The objective function evaluates or estimates certain criteria in a problem with respect to the solutions. This objective function can be used to estimate the quality of the solutions.

A single objective function is denoted as $f \in$ R. However, in MOO problem, there exist $M$-numbers of objective function, which is denoted as $f_j \in$ R for $j = 1,2,...,M$. Hence, Eq. (6) defines the vector of objective functions with respect to a solution, $x$.

$$\vec{f}(\vec{x}) = \{f_1(\vec{x}), f_2(\vec{x}), \ldots, f_M(\vec{x})\} \tag{6}$$

Fig. 1 illustrates the relationship between a solution and its mapped vector of objective functions. The search space contains all possible solutions and their mapped vector of objective functions are in the objective space.

### D. Pareto Dominance and Optimality

The conflicting objective functions in a MOO problem can lead to multiple incomparable solutions. For example, one solution is good for one objective but bad for other objectives compared to another solution. Thus, this section defines these incomparable solutions, which are equally important as they provide more option for decision making.

**Definition 4.** Consider $\vec{u} = \{u_1,u_2,...,u_M\}$ and $\vec{v} = \{v_1,v_2,...,v_M\}$ as two vectors where $\vec{u}$ dominates $\vec{v}$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if $u_j \leq v_j$ for every $j = \{1,2,...,M\}$ and $u_j < v_j$ for at least once.

Hence, there exist three possible relations between solutions under the Pareto dominance.

- $\vec{u}$ strictly dominates $\vec{v}$ ($\vec{u} \prec \vec{v}$) if and only if $u_j < v_j$ for every $j = \{1,2,...,M\}$

- $\vec{u}$ weakly dominates $\vec{v}$ ($\vec{u} \preceq \vec{v}$) if and only if $u_j \leq v_j$ for every $j$ ={1,2,...,$M$}
- $\vec{u}$ is incomparable to $\vec{v}$ ($\vec{u}$ □ $\vec{v}$) if and only if $u_j \nleq v_j$ and $u_j \ngeq v_j$ for every $j$ = {1,2,...,$M$}. This incomparable relation also known as *non-dominated* as the two vectors do not dominate each other.

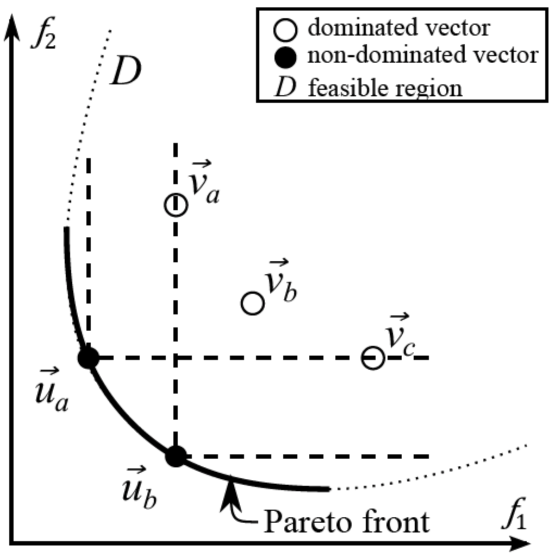
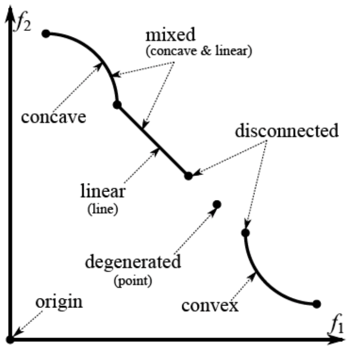
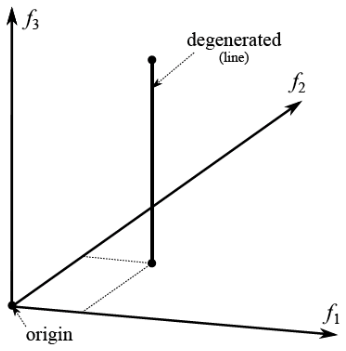


Fig. 2 Pareto Dominance relation for problem with two objective functions.



(a)



(b)

Fig. 3 Pareto front with different shapes for (a) two objectives problem and (b) three objectives problem.

Figure 2 illustrates the three dominance relations with two objective functions.

**Definition 5**. A solution, $\vec{x}^*$ ∈ F is considered as a *Pareto optimal* solution when there is no other solution $\vec{x}^*$ ∈F such that $\vec{f}(\vec{x}^*) \preceq \vec{f}(\vec{x})$.

This Pareto optimal solution is illustrated as $\vec{u}_a$ and $\vec{u}_b$ in Fig. 2, where they dominate the rest of the solutions. It is possible that two or more Pareto optimal solutions to be incomparable. Thus, these incomparable solutions resulted in multiple non-dominated solutions such as $\vec{u}_a$ and $\vec{u}_b$.

**Definition 6**. The group of non-dominated solutions in the search space is known as the Pareto optimal set, P.

The Pareto optimal set is illustrated as the set *E* in Fig. 1. However, there is a possibility that there are multiple Pareto optimal set scattered inside the feasible solution set *C*.

## *E. Pareto Front*

**Definition 7**. A set which consists all vector of objective functions that correspond to the P is known as the Pareto front, PF$= \{\vec{f}(\vec{x}) \mid \vec{x} \in \mathcal{P}\}$.

The Pareto front for a two objectives problem is illustrated in Fig. 1 and Fig. 2 by assuming that there are multiple non-dominated solutions distributed on the bold line. The main interest of MOO problems is to find the Pareto front as it shows the best feasible solution which can be used for decision making.

The Pareto front of a MOO problem is largely depended on the different features comprehended in the problem. These different features determine the shape and how the non-dominated solutions are mapped to the Pareto front. The combination of different features can increase the difficulty for an MOO algorithm to obtain the Pareto front that is similar to the actual solutions for the MOO problem. Several commonly found features in MOO problems are discussed in this work.

Some MOO problems have Pareto front with different shapes. This is illustrated in Fig. 3(a). The most common is the convex shape, where the objective fitness form a surface or boundary curves toward the origin. In contrast, the Pareto front also can have concave shape when the surface or boundary of the objective fitness curves outward from the origin. The Pareto front with linear shape occurs when the objective fitness form a flat line in the objective space.

In addition to the above, some MOO problems have Pareto front with mixed shapes. This refers to the Pareto front, which is a combination of different shapes mentioned earlier. An example of a simple Pareto front which comprehends the concave and linear shape. Usually, the objective fitness exist close to each other and form a continuous line or surface in the objective space. However, there is an instance where these objective fitness are broken into several continuous lines or surfaces. Such situation represents the Pareto front with discontinued shape.

In a few occasions, the MOO problem could also have Pareto front with degenerated shape when the dimension of the front is lower than the number of objective functions. For example, in a MOO problem with two objective functions, its Pareto front is only a point in objective space as shown in Fig. 3(a). Similarly, the front for a three objectives problem is just a line in the objective space as illustrated in Fig. 3(b).

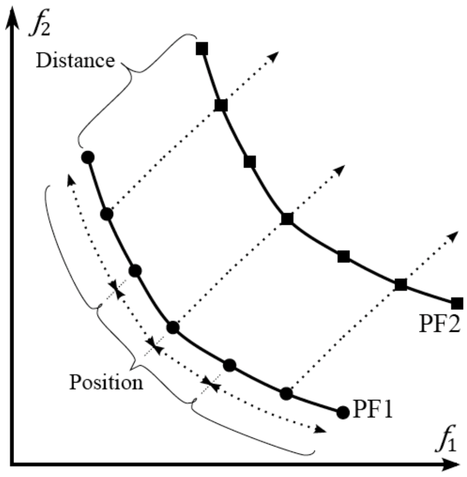


Fig. 4 Distance and position in Pareto fronts.

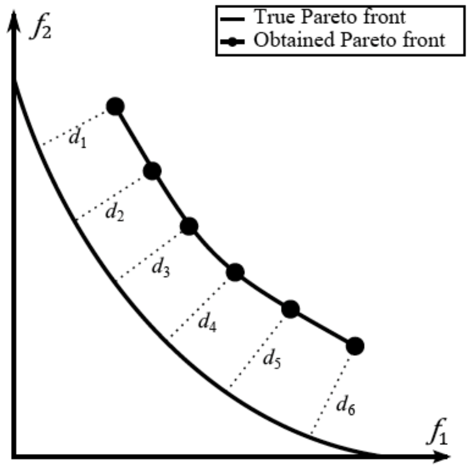


Fig. 5 Generational distance.

Some variations of distribution can be expected between an evenly distributed sample of non-dominated solutions in the search space and their evenly distributed set of objective fitness in the objective space. However, the Pareto front of some MOO problems may bias when the variation is significant. Thus, algorithm that operates with the guidance of the non-dominated solutions can be easily tricked by those biased solutions in the search space as they mapped differently in the objective space.

A MOO problem with one local optima solution is known as uni-modal problem. This type of problem is relatively simple because convergence at this solution is highly desirable. Conversely, an algorithm is unable to find all the feasible solutions if it has converged to one of the local optima. This MOO problem with multi local optima solutions is known as multi-modal problem.

It is possible to have different solutions in the search space which mapped to the same objective fitness in objective space. This situation is referred to as Pareto many-to-one feature and could increase the difficulty of a problem. Thus, it is hard for an algorithm to decide which solution to be maintained as all these solutions are equal in objective fitness.

## III. Performance Measure for Multi-objective Optimization Algorithms

Different features of MOO problem can affect the performance of MOO algorithm. Hence, performance measure is needed to determine how good an algorithm is. Performance measure can either be in the form of qualitative (i.e. graphically) or quantitative (i.e. mathematically). However, sometimes qualitatively evaluation could face incomparable situation. Thus, quantitative comparison is preferable because the quality of the solutions obtained by the algorithm is evaluated into a scalar value.

There exist various quantitative performance measures. However, in this work, several considerations are commonly emphasized in order to differentiate the performance of different algorithms [2-3]:

- Distance between the Pareto front obtained from the algorithm and the true Pareto front of a problem should be minimized, if the true Pareto front is known.
- The Pareto front obtained should be well distributed, where uniform distribution is favourable in most cases.
- Maximize the extent of the obtained Pareto front such that it could cover each objective as wide as possible.

Thus, several quantitative performance measures have been used to compare the performance of different algorithms by considering the convergence, diversity and the numbers of the non-dominated solutions. The convergence means the distance between two Pareto fronts, which is illustrated as the "Distance" in Fig. 4. Meanwhile, in the same figure, the diversity represents the distribution of the non-dominated solutions, which is expressed as the "Position".

### A. Number of Solution (NS)

The *NS* calculates the total number of non-dominated solutions obtained by the algorithm at the end of the iteration. Thus, the best algorithm gives the largest *NS* value by this measure. Its mathematical representation is defined as Eq. (7).

$$NS = | \mathcal{PF}_o | \tag{7}$$

### B. Generational Distance (GD)

The *GD* [4] is a popular measure for convergence performance [5-7]. This measure represents the average distance between the true Pareto front and the one obtained by the algorithm, as illustrated in Fig. 5. Consider a true Pareto front with *p*-number of non-dominated solutions, Eq. (8) computes the *GD*, with a smaller value corresponding to a better performance.

$$GD = \frac{(\sum_{i=1}^{n} d_i^{M})^{\frac{1}{M}}}{n} \tag{8}$$

where *n* is the number of non-dominated solutions obtained by the algorithm when solving a problem with *M* objective functions. The $d_i$ represents the minimum distance of a non-dominated solution from the true Pareto front, as illustrated in Fig. 5 and it is calculated using Eq. (9).

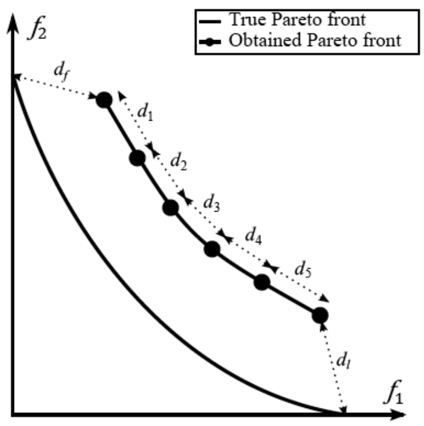


Fig. 6 Spread measurement.

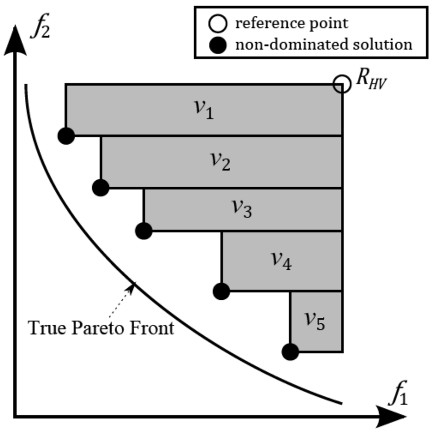


Fig. 7 Hypervolume measure with area covered by non-dominated solutions and reference point.

$$d_i - \min_{1\le k\le p} \sqrt{\sum_{j=1}^{M} (f_{i,j} - f_{i,k})^2} \tag{9}$$

## C. Spread (SP)

The *SP* [5] is commonly used to measure the diversity or the distribution of non-dominated solutions [8, 6]. This measure calculates the uniformity of the non-dominated solutions by using Eq. 10.

$$SP = \frac{d_f - d_l + \sum_{i=1}^{n-1} | d_i - \bar{d} |}{d_f + d_l + \bar{d}(n-1)} \tag{10}$$

where $d_f$ is the Euclidian distances between the first boundary solution of the non-dominated solutions obtained by the algorithm and the first boundary solution of true Pareto front, similarly for $d_l$ with the last boundary solution. The first and last boundary solutions are the first and last solutions of a set of non-dominated solutions, which has been sorted ascending at the first objective function. The $d_i$ is the Euclidean distance between one solution and the next solution. The $d_f$, $d_l$, and $d_i$ are illustrated in Fig. 6. Meanwhile, $\bar{d}$ is the mean of $d_i$ for $i = 1,2,...,n-1$ by assuming there are *n* numbers of non-dominated solutions obtained by the algorithm. Thus, *SP* actually measures the average distance between consecutive solutions for all non-dominated solutions obtained by the algorithm as well as two boundary solutions in the true Pareto front. Hence, the Pareto front with small *SP* value indicates that the non-dominated solutions have better distribution on the Pareto front.

## C. Hypervolume (HV)

The *HV* was proposed by Zitzler and Thiele [9]. This measure is used to evaluate the area (in a two-objective problem) or the volume between a reference point, $R_{HV}$ and the Pareto front with respect to the non-dominated solutions obtained by the algorithm. This is shown in Fig. 7 where the shaded area is the total area for *HV*. In order to have good convergence and diversity performance, the obtained non-dominated solutions should result in more distance from the reference point and equally distributed between the boundary solutions and the true Pareto front. In other words, it is desirable that the Pareto front obtained by the algorithm produces a large *HV* value.

In mathematical representation, the *HV* can be calculated using Eq. (11).

$$HV = \sum_{i=1}^{|\mathcal{PF}_o|} v_i \tag{11}$$

where $R_{HV}$ is the vector constructed from the worst objective value of the true Pareto front and $v_i$ is the hypercube constructed between $R_{HV}$ and the diagonal corner of the *i*-th solutions by assuming that there are *n* numbers of non-dominated solutions obtained.

## IV. Test Problem

Since different features in MOO problems are responsible for decreasing the likelihood of obtaining both convergence and diversity [2], standard test functions with well-defined true Pareto fronts are important to test optimization algorithms. Thus, researchers have introduced different MOO problems with different problem features.

Among various test problems, two sets of standard test problems, ZDT [3] and WFG [10], have been selected to test the MOO algorithms throughout this work. The ZDT test problem is commonly used for this purpose. Meanwhile, the WFG test problem covers wide variety and difficult problems with different features. Besides, both sets of test problem have well-defined true Pareto fronts, which are required when evaluating different performance measures. The true Pareto fronts for ZDT and WFG test problems is obtained from the standard database generated by jMetal (http://jmetal.sourceforge.net/problems.html).

### A. ZDT Test Problem

The ZDT test problem [3] consists of six variation of problems. However, the ZDT5, which is designed for binary evaluation, has been excluded as this study focuses on the continuous search space problem. The remaining ZDT problems have two objective functions and structured similarly as below:

$$\begin{aligned} &\text{Minimize} \vec{f}(\vec{x}) = \{f_1(x_1), f_2(\vec{x})\} \\ &\quad \text{where} f_2(\vec{x}) = g(x_2, x_3, \ldots, x_n) \cdot h(f_1(x_1), g(x_2, x_3, \ldots, x_n)) \end{aligned} \tag{12}$$

The detail formulations of each test problem can be found in [3].

The ZDT1 test problem examines algorithm's ability in handling MOO problem with convex shape Pareto front. This problem is relatively simple and its true Pareto front is plotted in Fig. 8. On the other hand, the ZDT2 examines concave shape Pareto front and its true Pareto front is plotted as in Fig. 9. However, the ZDT3 is much more complex as it is discontinued in both Pareto Optimal set and their Pareto front. In addition, the ZDT3 is mixed with convex shapes Pareto front, as in Fig. 10. Interestingly, in Fig. 11, the Pareto front of ZDT4 looks similar to the ZDT1, where both are in convex shapes. However, there are multi local optima solutions or multi-modality in ZDT4. Examples of the two local optima Pareto fronts are shown in Fig. 11. Finally, the ZDT6 test problem has two difficulties due to the non-uniformity of the search space. First, the non-dominated solutions are biased when $f_1(x_1)$ is nearer to one as shown in Fig. 12. Second, the feasible solutions are denser when it is far away from the true Pareto front and less dense when it is near to the front.

### B. WFG Test Problem

Other than ZDT test problems, WFG test problem [10] were also usually used to validate the performance of the algorithm. The plot of Pareto front for each WFG test problem with two objective functions are shown in Fig. 13-21.

## V. Overview of Multi-objective Particle Swarm Optimization Algorithms

The process of the PSO algorithm has been summarised in Algorithm 1. The initial position become the *pBest* when the position of particles is randomized and velocity of particles is set to zero, which results in *gBest* being the best among all *pBest*. Subsequently, the algorithm iterates by first updating the velocity, position and *pBest* for all particles followed by updating the *gBest* and evaluating the termination criterion. Note that the algorithm iterates until the termination criterion is met. This termination criterion can either be the maximum number of iteration or if the fitness of the *gBest* is better than a desired fitness value.

Initially, the PSO algorithm is devoted for single objective optimisation only. However, researchers have extended the PSO algorithm to various Multi-objective Particle Swarm Optimization algorithms for solving MOO problems. Multi-objective Particle Swarm Optimization algorithms can be categorised into six different approaches as shown in Fig. 22. The categorisation is based on the approach used to collect and select the solution that guide particles flight, known as leader particle. Then, algorithms under Pareto-based approach are further categorised based on the additional selection mechanism used in these algorithms.

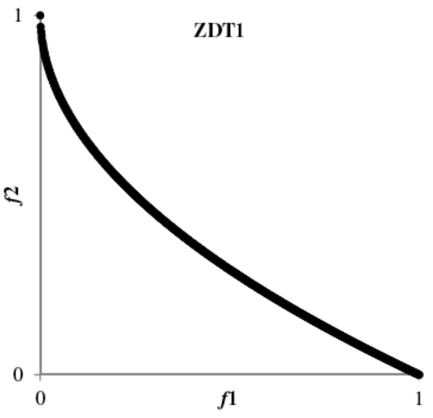


Fig. 8 True Pareto Front for ZDT1.

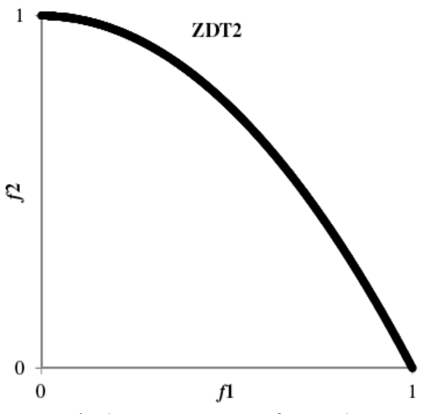


Fig. 9 True Pareto Front for ZDT2.

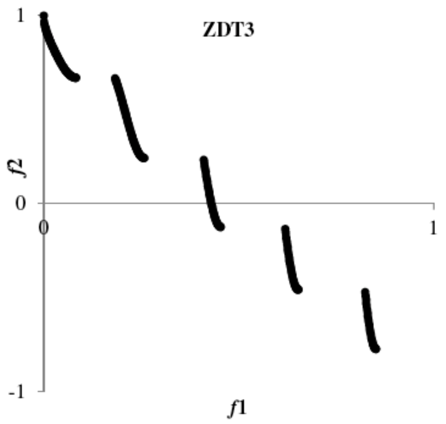


Fig. 10 True Pareto Front for ZDT3.

It is important to mention that in conventional PSO, movement of particles in search space towards optimum solution are influenced by its previous condition, its Personal best-found solution (*pBest*) and the best-found solution among all particles (*gBest*). The *pBest* of a particle is the solution that is best at a specific objective function among all solutions.

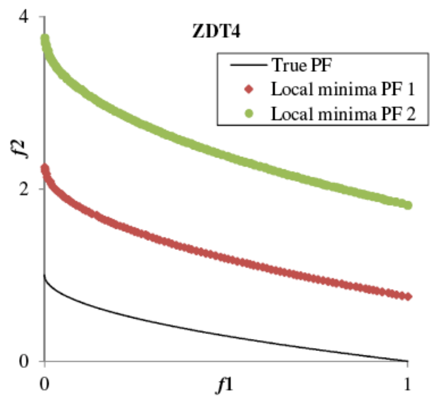


Fig. 11 True Pareto Front for ZDT4.

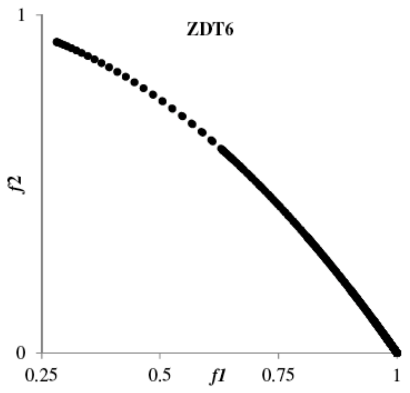


Fig. 12 True Pareto Front for ZDT6.

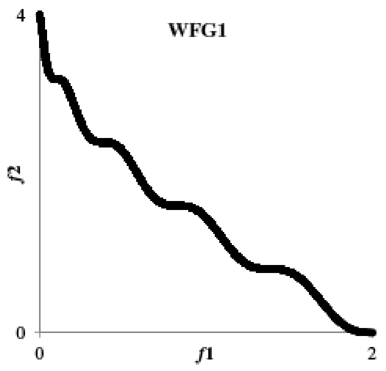


Fig. 13 True Pareto Front for WFG1.

Meanwhile, the *gBest* is the best among all *pBest* of all particles at the specific objective function. However, to solve MOO problems with PSO, an archive is usually used to keep track on the non-dominated solutions found during computation based on the concept of Pareto Dominance. At the end of computation, the solutions in archive represent the possible best solution for the MOO problem.

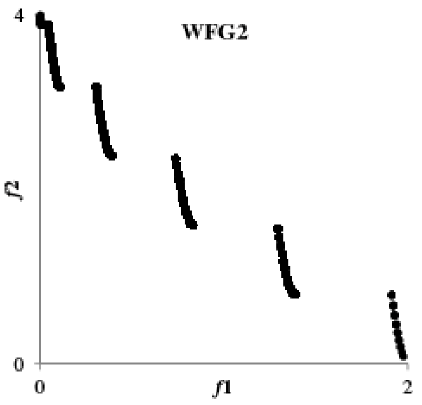


Fig. 14 True Pareto Front for WFG2.

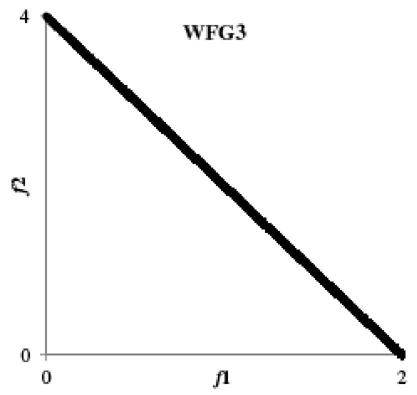


Fig. 15 True Pareto Front for WFG3.

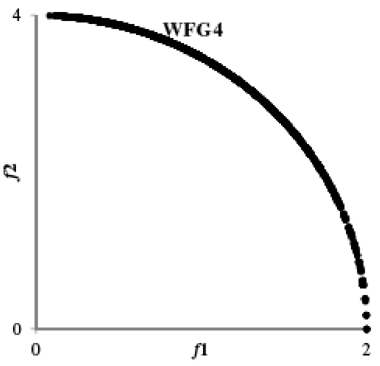


Fig. 16 True Pareto Front for WFG4.

## A. Aggregation Approach

In this approach, multiple objective functions from the MOO problems are composed (or "aggregated") into a single objective function. During computation, the solution with best objective fitness is selected as the leader to guide the search for optimum solution.

Three different algorithms which aggregate the objective functions to a weighted combination could be found in the work of Parsopoulos and Vrahatis [12]. The first algorithm using fixed weights in computation, known as Conventional Weighted Aggregation (CWA) was unable to solve problem with Pareto front of concave geometry [13]. Therefore, the second algorithm used weights that changed abruptly, known as Bang-Bang Weighted Aggregation (BWA). In addition, the weights are gradually changed from time to time in the third algorithm, known as Dynamic Weighted Aggregation (DWA) allow optimum solutions to be found along the Pareto fronts. All three algorithms recorded the non-dominated solutions in an external archive during computations, where these non-dominated solutions are selected based on the concept of Pareto dominance.

Similarly, in [14], the CWA approach is used as its major mechanism. However, the author divided the swarm into several sub-populations, where each sub-population has its own set of weighted sum and its own leader for guidance. During computation, the gradient of the objective fitness is used to identify the non-dominated solutions.

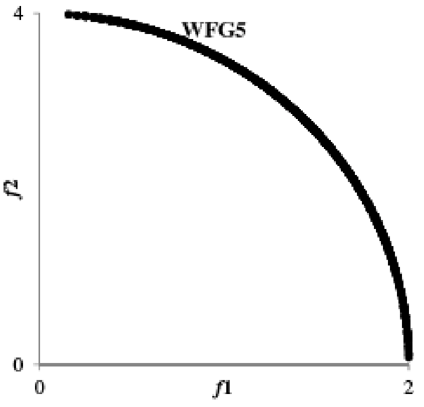


Fig. 17 True Pareto Front for WFG5.

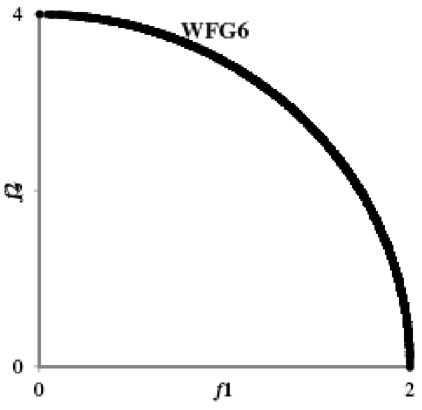


Fig. 18 True Pareto Front for WFG6.

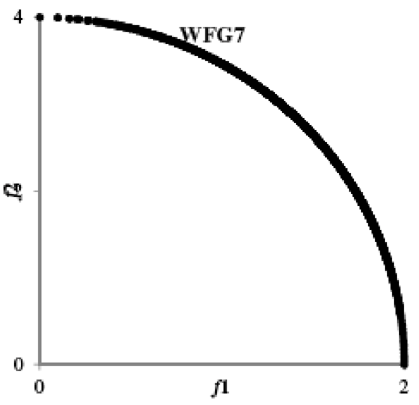


Fig. 19 True Pareto Front for WFG7.

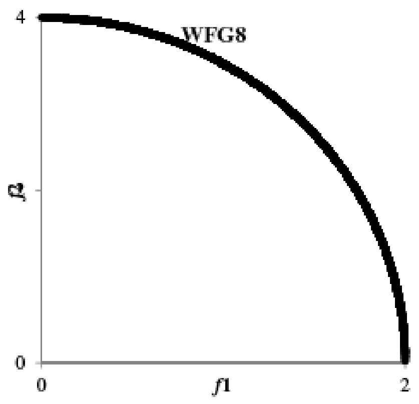


Fig. 20 True Pareto Front for WFG8.

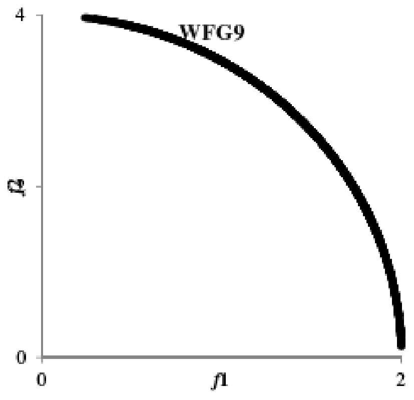


Fig. 21 True Pareto Front for WFG9.

Aggregation approach is relatively simple as there is only one objective function to compute. However, this approach suffers some drawbacks, where all objective functions are compromised when aggregated into single objective function, especially when there is no proper weights is acknowledged.

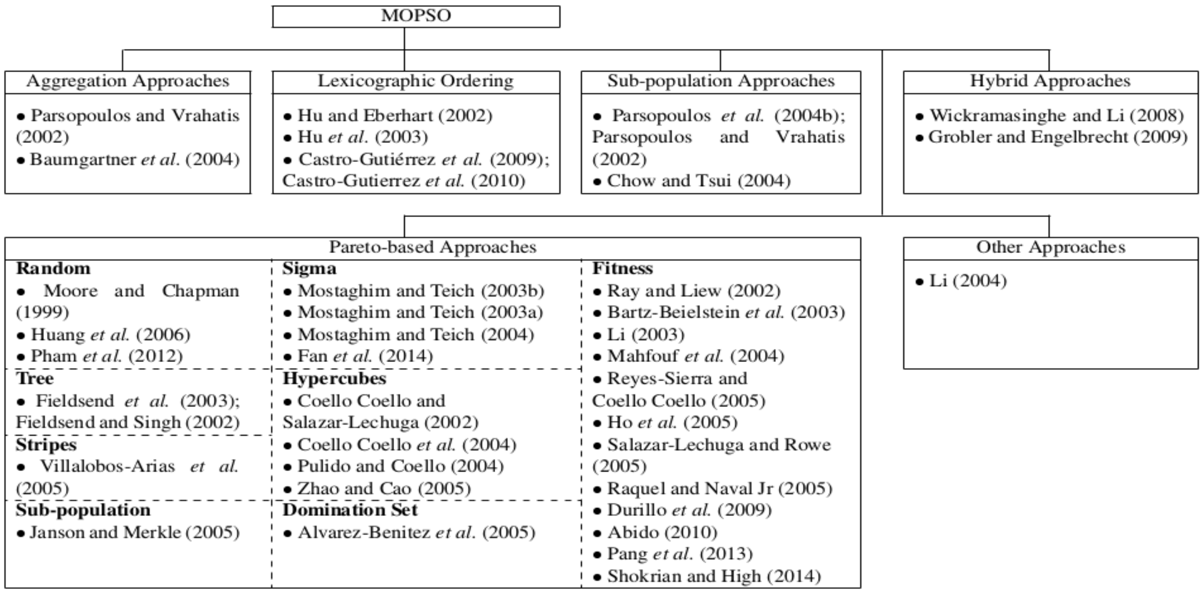

MOPSO
Aggregation Approaches
• Parsopoulos and Vrahatis (2002)
• Baumgartner et al. (2004)
Lexicographic Ordering
• Hu and Eberhart (2002)
• Hu et al. (2003)
• Castro-Gutiérrez et al. (2009); Castro-Gutierrez et al. (2010)
Sub-population Approaches
• Parsopoulos et al. (2004b); Parsopoulos and Vrahatis (2002)
• Chow and Tsui (2004)
Hybrid Approaches
• Wickramasinghe and Li (2008)
• Grobler and Engelbrecht (2009)
Pareto-based Approaches
Random
• Moore and Chapman (1999)
• Huang et al. (2006)
• Pham et al. (2012)
Tree
• Fieldsend et al. (2003); Fieldsend and Singh (2002)
Stripes
• Villalobos-Arias et al. (2005)
Sub-population
• Janson and Merkle (2005)
Sigma
• Mostaghim and Teich (2003b)
• Mostaghim and Teich (2003a)
• Mostaghim and Teich (2004)
• Fan et al. (2014)
Hypercubes
• Coello Coello and Salazar-Lechuga (2002)
• Coello Coello et al. (2004)
• Pulido and Coello (2004)
• Zhao and Cao (2005)
Domination Set
• Alvarez-Benitez et al. (2005)
Fitness
• Ray and Liew (2002)
• Bartz-Beielstein et al. (2003)
• Li (2003)
• Mahfouf et al. (2004)
• Reyes-Sierra and Coello Coello (2005)
• Ho et al. (2005)
• Salazar-Lechuga and Rowe (2005)
• Raquel and Naval Jr (2005)
• Durillo et al. (2009)
• Abido (2010)
• Pang et al. (2013)
• Shokrian and High (2014)
Other Approaches
• Li (2004)

Fig. 22 Categories of various MOPSO algorithms.

The algorithm reported in [15] is considered as Lexicographical ordering because the first objective fitness are fixed (most important) while being optimised by the second objective function (least important), but the authors did not clarify how to fix the first objective fitness. The authors claimed the first objective fitness is fixed so the solutions will “dropped” down onto the Pareto front. Therefore, external archive has not been used. However, this algorithm has been improved with external archive for this purpose [16], yet they did not clarify the method for fixing the first objective fitness. Later, this approach is useful only with small number of objective functions because of the sensitivity in the order of importance on all objective functions.

Dynamic Lexicographical ordering was used in [17-18]. In contrast to traditional Lexicographic ordering, the priority of the objective functions is dynamically modified during the computation.

## B. *Sub-population Approach*

Each objective function in MOO problem is optimised by a sub-population under a SOO algorithm. The leader for each sub-population is the solution with the best objective fitness at the respective objective function. Besides, these sub-populations collaborate by exchanging information in term of solution optimised for individual objective function and aimed to obtain the trade-off solutions which satisfy all objective functions.

Using sub-population approaches, Vector Evaluated Particle Swarm Optimization (VEPSO) algorithm has been introduced [19]. VEPSO is inspired by the Vector Evaluated Genetic Algorithm (VEGA) [20-21]. In VEPSO algorithm, each sub-population optimises a single objective function with the guidance from the best-found solution of another sub-population. The guidance from another sub-population encourages the algorithm to obtain the solutions considering the trade-off between all objective functions and lead to the Pareto front.

In [22], a PSO algorithm is extended to the Multi-Species Particle Swarm Optimization (MS-PSO) algorithm for autonomous agent response learning. In MS-PSO, each species is a sub-population that optimises a specific objective function. Thus, the number of species is equal to the number of objective function. Besides, a communication channel is established between all species by sharing their best-found solution to serve as the guidance in optimising the objective functions. This communication is the sum of difference between a particle of one species to the best-found solutions of all species.

### C. Pareto-based Approach

Pareto Dominance is the main mechanism employed in this approach to collect all non-dominated solutions in an archive. Then, the leader for guiding the search is selected from the non-dominated solutions of their respective swarm. Therefore, Pareto Dominance directly influences the selected leader. However, a number of different selection strategies are available among those equally good non-dominated solutions.

In [23], PSO is modified with the concept of Pareto dominance for solving MOO problems. Unlike the conventional PSO, the *pBest* of a particle is randomly selected from a list of non-dominated solutions found by that particle only. Meanwhile, the *gBest* is a solution that dominates all solutions in the list of all particles. However, there is no information on how to select the *gBest* when there is more than one solutions that dominate the rest of the solutions.

Various mechanisms have been used in [24]. First, the authors used Pareto ranking and multi-sieve strategy to generate a set of leaders, which contains only the non-dominated solutions [25]. Then, those particles that are not in the set of leaders are subjected to select a leader. This leader selection is based on roulette wheel selection, where the non-dominated solution with larger crowding radius (fewer solutions around it) has higher probability to be selected to maintain spread along Pareto front by exploring new areas. Lastly, simple generational operator [26] is used to obtain information from the leader, which yield a unique position for particle at neither its original position nor its leader position and prevent premature convergence.

Fieldsend and Singh [27] and Fieldsend *et al.* [28] utilized the dominate trees data structure [29] to select a leader for each particle from global archive. Each particle chooses one of the nearest members (in objective space) from the tree of the composite point, which is located based on the dominance relation.

Coello Coello and Salazar-Lechuga [30] and Coello Coello *et al.* [31] had proposed a famous MOPSO, where the *pBest* is the latest solution that is not dominated by the previous *pBest*. Besides, all non-dominated solutions are stored in a global archive and are divided into several hypercubes. The particles leader is chosen randomly among the non-dominated solutions from a hypercube. Pulido and Coello [7] have further improved this algorithm using clustering technique, where population is divided into several sub-population for better distribution of decision space. Besides, the selection pressure of leader is periodically varied by migrating leaders of one sub-population to another during the search process.

Mostaghim and Teich [32] proposed a Sigma MOPSO algorithm, which assigned all particles and non-dominated solutions (in global archive) with a sigma value. When updating the velocity of the particles, the leader for each particle was selected based on one of the closest non-dominated solutions in term of sigma values. This algorithm used clustering method [9] to manage the number of solutions in an archive. However, the authors found that the computation of this algorithm is more efficient when its archive is managed by the dominance technique [33]. Besides, the authors also improved the Sigma MOPSO into a two stages algorithm using sub-populations [34]. In this two steps algorithm, the non-dominated solutions obtained from the Sigma MOPSO are used as the leaders for the second stage. In the second stage, each leader is used to fine-tune the Pareto front by each sub-population.

Bartz-Beielstein *et al.* [35] introduced elitism with the use of archive into MOPSO algorithm called DOPS . Each particle is assigned with a selection fitness and deletion fitness, which are evaluated by different selection functions (related to the number of successive selection or the fitness of the solution) and deletion function (related to the diversity of the Pareto front) respectively. Therefore, the *pBest* and leader could be selected from an archive of non-dominated solutions based on roulette wheel selection. Meanwhile, solutions would be removed based on the deletion fitness when the archive exceeds its limited size.

On the other hand, Li [36] proposed the Non-dominated Sort Particle Swarm Optimization (NSPSO) by incorporating the main mechanism in Non-dominated Sort Genetic Algorithm-II [5] into MOPSO. During computation, a new population of particles is formed by selecting the non-dominated solutions (by means of non-dominating sort) from the combination of all *pBest* and newly generated solutions. This new population will be used to generate a new solution. During this process, each particle is guided by a leader from an external archive, which is randomly selected by means of niche count or crowding distance value (for NSPSO with niche count or NSPSO with crowding distance respectively).

Furthermore, Reyes-Sierra and Coello Coello [37] also developed a MOPSO algorithm based on Pareto dominance, denoted as Multi Objective Particle Swarm Optimization (OMOPSO). Two non-dominated solutions are randomly selected from archive and are subjected to binary tournament based on their crowding distance value to determine the leader to guide a particle. The winner of the tournament is assigned as the leader. The authors also used the crowding distance value to filter the non-dominated solutions in archive when the archive is full. Meanwhile, particles undergo position mutation and non-dominated solutions are maintained by dominance in order to yield better performance.

Meanwhile, Alvarez-Benitez *et al.* [38] proposed three different selection techniques, which are called ROUNDS, RANDOM, and PROB. ROUNDS technique focuses on promoting diversity. The non-dominated solutions that dominate the least number of particles become the leader for those particles and this selection is applied until all particles have their own leader. RANDOM technique promotes convergence. The leader of a particle is randomly selected from a set of non-dominated solutions that has dominated that particle. Lastly, PROB is a weighted probabilistic, which form a compromise of both ROUNDS and RANDOM in order to obtain Pareto front with a balance of convergence and diversity.

A MOO algorithm introduced by Ho *et al.* [39] considers an archive for each particle to keep track the non-dominated solutions found and for selecting its *pBest*. Another global archive is used to keep track the non-dominated solutions found by all particles. The strength fitness [9] is used for selecting *pBest* and leader for particles. The strength fitness for each non-dominated solution is evaluated based on the number of particles dominated by that solution. Besides, each non-dominated solution has an age fitness, which increases during computation. It is used in parallel with the strength fitness during the selection process. Both the strength and age fitness promote diversity for the distribution of non-dominated solutions along the Pareto front.

Villalobos-Arias *et al.* [40] used a new mechanism for leader selection, which is called stripes. The non-dominated solutions are used to divide the objective space into several stripes. Each non-dominated solution becomes the centre of gravity for that stripe. Then, each particle selects a non-dominated solution correspond to the stripe as its leader.

Another MOO algorithm, which emphasizes the distribution of solutions along the Pareto front was introduced by Salazar-Lechuga and Rowe [41]. This algorithm stores all non-dominated solutions in an external archive. Each non-dominated solution is assigned with a fitness value. This fitness value is evaluated by the niche count function [42], which indicates the density around a solution. Solution with less dense neighbour has higher tendency to be selected as leader. Besides, this niche count fitness also very important for managing the number of solutions in the archive, where solution with the most dense neighbouring will be removed if the limit of archive is exceeded.

Raquel and Naval Jr [43] also proposed a MOO algorithm which uses crowding distance value. Similar to other algorithms, the crowding distance value is used for selecting a leader and for deleting a solution when the archive limit has exceeded. Besides, mutation mechanism [31] is also included in this algorithm for preventing premature convergence.

The algorithm proposed by Zhao and Cao [44] is very similar to [41], where hypercubes are used for leader selection and deletion of excess solutions in archive. However, this algorithm has two archives, where one stores the non-dominated solutions found during computation and another stores the best *pBest*.

A MOPSO algorithm with multiple swarms is introduced by Janson and Merkle [45]. This algorithm is called ClustMPSO, where *K*-mean algorithm is used to cluster the particles into several sub-populations. Each sub-population has its own archive, which records all non-dominated solutions encountered by this sub-population. A randomly selected solution from the archive of a swarm is used to guide all particles in that swarm. This selected solution will only be re-selected after a few iterations or if it is no longer a non-dominated solution. The combination of archive from all sub-populations yields a global archive, which only kept those solutions that are truly non-dominated.

Durillo *et al.* [8] analyzed OMOPSO with various MOPSO algorithms and proposed a modified OMOPSO called SMPSO by including a velocity constraint mechanism. Similarly, SMPSO also updates particles velocity using only one leader based on a suitable crowding distance. However, the velocities are clamped to a certain limit according to the search space dimension.

Abido published a new approach, which used two set of non-dominated solutions, global for whole swarm and local for each particle [46]. The *pBest* and the leader of each particle are selected based on the distance between the particle and the members in its respective non-dominated solution set. The non-dominated solution with the shortest distance in objective space from the particle will be selected as a leader.

Recently, Pham *et al.* [47] introduced the use of multi-guider (also known as leader) for new MOPSO algorithm. However, there are only two leaders used in this new approach. For a swarm of particles, their first leader is randomly selected from the top percentage of the non-dominated solutions, which were sorted by crowding distance. The second leader is the closest non-dominated solution from the first leader in the objective space.

Pang *et al.* [48] proposed a new MOPSO, which relies on entropy-based density assessment strategy and adaptive chaotic mutation operator, namely MOQPSO-AE. The information contributed by each non-dominated solutions is quantified using the concept of entropy information gain [49]. Hence, each non-dominated solution is assigned with an entropy probability evaluate using Parzen window density estimator. The entropy probability is higher when solutions are uniformly distributed along Pareto front. Thus, non-dominated solution with higher entropy probability is more likely to be selected as leader using roulette wheel selection strategy. Besides, this entropy probability is used for adaptive mutation strategy, where a percent of solutions in population with lowest entropy probability are mutated. Chaos is natural non-linear phenomena, which is characterised by ergodicity, randomicity and sensitivity to initial conditions. Hence, chaos iteration logistic function, which has more search probability at boundary of search space than Gaussian distribution is used for mutating the position of solutions with low entropy probability.

Another recent work by Shokrian and High [50] also introduce another algorithm, which use multi-leaders for guiding particles. Particle movement is guided by a fixed number of leaders but the amplitude of their influence depends on their distance to that particle and a Social Influence Factor (SIF). Smaller SIF reduces the influence of leaders to only those closest to a particle and larger SIF influences leader that are far from a particle. Additionally, archive is used to store all non-dominated solutions found, where these solutions are assigned with aggregated fitness value. Non-dominated solutions are sorted based on this fitness value such that those solutions at the top are selected into the archive while those at the bottom are deleted when archive size is exceeded. Adaptive grid by [30, 51] was used to separate non-dominated solutions into hypercubes. Meanwhile, aggregated fitness of a solution is weighted summation of the deletion function (type 1 and 2 from [35]) and the normalised frequency of a solution is being selected as leader. However, the SIF and number of leaders is a tuning parameter so either expert knowledge or a fine tune is required to obtain good results.

Fan *et al.* [52] introduced new Pareto-based MOPSO algorithm called Empirical Movement and Diversified Search Multi Objective Particle Swarm Optimisation (EMDS-MOPSO). This algorithm combines two leader selection methods as diversified search strategy. If there is no individual in archive is dominated by other solution in a population, an individual, which is farthest from a centroid of archive, is selected as leader for the next generation. In another case, a random leader is selected based on sigma calculation of [32]. Additionally, authors also introduced a novel picking rule for maintaining only diverse solutions in archive. Initially, two most distanced solutions, in terms of their objective fitness are selected. Then, another unselected solution, which is most distanced from those selected solution is also selected and this process is repeated until the archive is full. Polynomial mutation [53] was also included to increase diversity performance.

### *D. Hybrid Approach*

Researches have proposed combination of different approaches. For example, a concept of selecting leader using Differential Evolution (DE) [54] in MOPSO algorithm was proposed by Wickramasinghe and Li [55]. Most processes in this algorithm are similar to MOPSO algorithm except for leader selection. During leader selection process, three particles are randomly selected to generate the offspring (leader) using DE. According to the authors, these three particles could come from different regions so that the generated leader could be positioned away from their local optimal solution. Therefore, this leader can prevent particles from trapping in a certain region and have premature convergence.

Meanwhile, Grobler and Engelbrecht [56] introduced the Vector Evaluated Differential Evolution Particle Swarm Optimisation (VEDEPSO), which combined the VEPSO and Vector Evaluated Differential Evolution (VEDE) [57]. For two objective problems, one population will optimise an objective using the velocity update in VEPSO algorithm and another population for another objective using updating process from VEDE. Additionally, best solutions from each population is exchanged with another population such that optimal solutions for these two objectives can be found.

### *E. Other Approach*

Those algorithms, which do not fit to any of mentioned approaches fall under this approach. Li proposed an algorithm called maximinPSO [58], which uses the scalar fitness value derived from maximin strategy [59] to obtain the non-dominated solutions. The leader for each particle is randomly selected from the non-dominated solutions based on the maximin fitness.

## VI. Conclusions

In this paper, MOO problem are clearly described, which includes the explanation for decision variables as the feasible solution, objective fitness as the values for objective functions and the constraint functions. In addition, various features which are commonly found in MOO problem are briefly explained for better understanding of the shapes, bias, modality, and separable in Pareto front. Several performance measures are also discussed. Then, two different sets of standard benchmark test problems, namely ZDT and WFG are explained

Importantly, various MOPSO algorithms have been reviewed. These algorithms are categorized into six different approach-based leader selections to guide the search process. Noticeably, most of the algorithms are based on Pareto-based approach. Furthermore, all reviewed algorithms are similar in a way that they only use one solution as the leader to guide the particles.

## Acknowledgment

This work is financially supported through Research Acculturation Grant Scheme (RDU121403) granted to Universiti Malaysia Pahang and Fundamental Research Grant Scheme (FRGS/1/2012/TK06/MMU/03/7) granted to Multimedia University. This research is also partly supported by the Ministry of Higher Education (MOHE) and Universiti Teknologi Malaysia through the Research University Grant (Q.J130000.2523.05H40).

## References

[1] Kennedy, J. and Eberhart, R. C. (1995). Particle Swarm Optimization. In Proceedings IEEE International Conference on Neural Networks, vol. 4. pp. 1942–1948.

[2] Deb, K. (1999). Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. Evolutionary computation. 7(3), pp. 205–230.

[3] Zitzler, E., Deb, K. and Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical results. Evolutionary Computation. 8(2), pp. 173–195.

[4] Van Veldhuizen, D. A. (1999). Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. Ph.D. Thesis. Air Force Institute of Technology, Air University.

[5] Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002a). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation. 6(2), pp. 182–197.

[6] Fan, S.-K. and Chang, J.-M. (2009). A parallel Particle Swarm Optimization Algorithm for Multi-objective Optimization Problems. Engineering Optimization. 41(7), pp. 673–697.

[7] Pulido, G. T. and Coello, C. A. C. (2004). Using Clustering Techniques to Improve the Performance of A Multi-objective Particle Swarm Optimizer. In Genetic and Evolutionary Computation–GECCO 2004. Springer, pp. 225–237.

[8] Durillo, J. J., Garc´ıa-Nieto, J., Nebro, A. J., Coello Coello, C. A., Luna, F. and Alba, E. (2009). Multi-objective Particle Swarm Optimizers: An experimental comparison. In Evolutionary Multi-Criterion Optimization. Springer, pp. 495–509.

[9] Zitzler, E. and Thiele, L. (1999). Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation. 3(4), pp. 257–271.

[10] Huband, S., Barone, L., While, L. and Hingston, P. (2005). A Scalable Multi-objective Test Problem Toolkit. In Evolutionary Multi-criterion Optimization. Springer, pp. 280–295.

[11] Huband, S., Hingston, P., Barone, L. and While, L. (2006). A Review of Multiobjective Test Problems and A Scalable Test Problem Toolkit. IEEE Transactions on Evolutionary Computation. 10(5), pp. 477–506.

[12] Parsopoulos, K. E. and Vrahatis, M. N. (2002a). Particle Swarm Optimization Method in Multiobjective Problems. In Proceedings of the 2002 ACM Symposium on Applied Computing. ACM, pp. 603–607.

[13] Jin, Y., Olhofer, M. and Sendhoff, B. (2001). Dynamic Weighted Aggregation for Evolutionary Multi-objective Optimization: Why Does It Work and How? In Proceedings of Genetic and Evolutionary Computation Conference. San Francisco, USA, pp. 1042–1049.

[14] Baumgartner, U., Magele, C. and Renhart, W. (2004). Pareto Optimality and Particle Swarm Optimization. IEEE Transactions on Magnetics. 40(2), pp. 1172–1175.

[15] Hu, X. and Eberhart, R. (2002). Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization. In Proceedings of the 2002 Congress on Evolutionary Computation (CEC–2002), vol. 2. IEEE Computer Society, pp. 1677–1681.

[16] Hu, X., Eberhart, R. and Shi, Y. (2003). Particle Swarm with Extended Memory for Multiobjective Optimization. In Proceedings of the 2003 IEEE Swarm Intelligence Symposium (SIS–2003). IEEE, pp. 193–197.

[17] Castro-Guti´errez, J., Landa-Silva, D. and Moreno Perez, J. (2009). Dynamic Lexicographic Approach for Heuristic Multi-objective Optimization. In Proceedings of the Workshop on Intelligent Metaheuristics for Logistic Planning (CAEPIA-TTIA 2009), Seville, Spain. pp. 153–163.

[18] Castro-Gutierrez, J., Landa-Silva, D. and P´erez, J. M. (2010). Improved Dynamic Lexicographic Ordering for Multi-objective Optimisation. In Parallel Problem Solving from Nature, PPSN XI. (pp. 31–40). Springer.

[19] Parsopoulos, K. E., Tasoulis, D. K., Vrahatis, M. N. et al. (2004). Multiobjective Optimization Using Parallel Vector Evaluated Particle Swarm Optimization. In Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA–2004), vol. 2. pp. 823–828.

[20] Sawaragi, Y., Nakayama, H. and Tanino, T. (1985). Theory of Multiobjective Optimization. vol. 176. Academic Press New York.

[21] Schaffer, J. D. (1984). Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms (aArtificial Intelligence, Optimization, Adaptation, Pattern Recognition). Ph.D. Thesis. Vanderbilt University, Nashville, TN, USA.

[22] Chow, C.-k. and Tsui, H.-t. (2004). Autonomous Agent Response Learning by A Multi-species Particle Swarm Optimization. In Congress on Evolutionary Computation, 2004. CEC2004., vol. 1. IEEE, pp. 778–785.

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

[24] Ray, T. and Liew, K. (2002). A Swarm Metaphor for Multiobjective Design Optimization. Engineering Optimization. 34(2), pp. 141–153.

[25] Ray, T., Kang, T. and Chye, S. K. (2000). An Evolutionary Algorithm for Constrained Optimization. In GECCO. pp. 771–777.

[26] Deb, K. and Kumar, A. (1995). Real-coded Genetic Algorhithms with Simulated Binary Crossover: Studies on Multimodel and Multiobjective Problems. Complex Systems. 9(6), pp. 431–454.

[27] Fieldsend, J. E. and Singh, S. (2002). A Multi-objective Algorithm Based Upon Particle Swarm Optimization, An Efficient Data Structure and Turbulence. In Workshop on Computational Intelligence. pp. 37–44.

[28] Fieldsend, J., Everson, R. and Singh, S. (2003). Using Unconstrained Elite Archives for Multiobjective Optimization. IEEE Transactions on Evolutionary Computation. 7(3), pp. 305–323.

[29] Everson, R., Fieldsend, J. E. and Singh, S. (2002). Full Elite Sets for Multi-Objective Optimisation. In Adaptive Computing in Design and Manufacture V. Springer, pp. 343–354.

[30] Coello Coello, C. A. and Salazar-Lechuga, M. (2002). MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In Proceedings of the 2002 Congress on Evolutionary Computation (CEC–2002), vol. 2. IEEE, pp. 1051–1056.

[31] Coello Coello, C. A., Pulido, G. T. and Salazar-Lechuga, M. (2004). Handling Multiple Objectives with Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation. 8(3), pp. 256–279.

[32] Mostaghim, S. and Teich, J. (2003b). Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization (MOPSO). In Proceedings of the 2003 IEEE Swarm Intelligence Symposium (SIS–2003). IEEE, pp. 26–33.

[33] Mostaghim, S. and Teich, J. (2003a). The Role of -dominance in Multi Objective Particle Swarm Optimization Methods. In The 2003 Congress on Evolutionary Computation (CEC–2003), vol. 3. IEEE, pp. 1764–1771.

[34] Mostaghim, S. and Teich, J. (2004). Covering Pareto-optimal Fronts by Subswarms in Multi-objective Particle Swarm Optimization. In Congress on Evolutionary Computation (CEC–2004), vol. 2. IEEE, pp. 1404–1411.

[35] Bartz-Beielstein, T., Limbourg, P., Mehnen, J., Schmitt, K., Parsopoulos, K. E. and Vrahatis, M. N. (2003). Particle Swarm Optimizers for Pareto Optimization with Enhanced Archiving Techniques. In The 2003 Congress on Evolutionary Computation, 2003. CEC'03., vol. 3. IEEE, pp. 1780–1787.

[36] Li, X. (2003). A Non-Dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. In Genetic and Evolutionary Computation (GECCO–2003). Springer, pp. 37–48.

[37] Reyes-Sierra, M. and Coello Coello, C. A. (2005). Improving PSO-Based Multiobjective Optimization Using Crowding, Mutation and -dominance. In Evolutionary Multi-Criterion Optimization. Springer, pp. 505–519.

[38] Alvarez-Benitez, J. E., Everson, R. M. and Fieldsend, J. E. (2005). A MOPSO Algorithm Based Exclusively on Pareto Dominance Concepts. In Evolutionary Multi-Criterion Optimization. Springer, pp. 459–473.

[39] Ho, S., Yang, S., Ni, G., Lo, E. and Wong, H. C. (2005). A Particle Swarm Optimization-based Method for Multiobjective Design Optimizations. IEEE Transactions on Magnetics. 41(5), pp. 1756–1759.

[40] Villalobos-Arias, M. A., Pulido, G. T. and Coello Coello, C. A. (2005). A Proposal to Use Stripes to Maintain Diversity in A Multi-objective Particle Swarm Optimizer. In Proceedings 2005 IEEE Swarm Intelligence Symposium (SIS–2005). IEEE, pp. 22–29.

[41] Salazar-Lechuga, M. and Rowe, J. E. (2005). Particle Swarm Optimization and Fitness Sharing to Solve Multi-objective Optimization Problems. In The 2005 IEEE Congress on Evolutionary Computation (CEC–2005), vol. 2. IEEE, pp. 1204–1211.
[42] Goldberg, D. and Richardson, J. (1987). Genetic Algorithms with Sharing for Multimodal Function Optimization. In Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application. Lawrence Erlbaum Associates, Inc., pp. 41–49.
[43] Raquel, C. R. and Naval Jr, P. C. (2005). An Effective Use of Crowding Distance in Multiobjective Particle Swarm Optimization. In Proceedings of the 2005 Conference on Genetic and Evolutionary Computation. ACM, pp. 257–264.
[44] Zhao, B. and Cao, Y.-j. (2005). Multiple Objective Particle Swarm Optimization Technique for Economic Load Dispatch. Journal of Zhejiang University SCIENCE A. 6(5), pp. 420–427.
[45] Janson, S. and Merkle, D. (2005). A New Multi-objective Particle Swarm Optimization Algorithm Using Clustering Applied to Automated Docking. In Blesa, M., Blum, C., Roli, A. and Sampels, M. (Eds.) Hybrid Metaheuristics. (pp. 128–141). Lecture Notes in Computer Science, vol. 3636. Springer Berlin Heidelberg.
[46] Abido, M. A. (2010). Multiobjective Particle Swarm Optimization with Nondominated Local and Global Sets. Natural Computing. 9(3), pp. 747–766.
[47] Pham, M.-T., Zhang, D. and Koh, C. S. (2012). Multi-Guider and Cross-Searching Approach in Multi-Objective Particle Swarm Optimization for Electromagnetic Problems. IEEE Transactions on Magnetics. 48(2), pp. 539–542.
[48] Pang, S., Zou, H., Yang, W. and Wang, Z. (2013). An Adaptive Mutated Multi-objective Particle Swarm Optimization with an Entropy-based Density Assessment Scheme. Information Computational Science. 4, pp. 1065–1074.
[49] Tan, K. C., Goh, C. K., Mamun, A. A. and Ei, E. Z. (2008). An evolutionary artificial immune system for multi-objective optimization. European Journal of Operational Research. 187(2), pp. 371–392.
[50] Shokrian, M. and High, K. A. (2014). Application of a multi objective multi-leader particle swarm optimization algorithm on NLP and MINLP problems. Computers and Chemical Engineering. 60, pp. 57–75.
[51] Knowles, J. D. and Corne, D. W. (2000a). Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation. 8(2), pp. 149–172.
[52] Fan, S.-K. S., Chang, J.-M. and Chuang, Y.-C. (2014). A new multi-objective particle swarm optimizer using empirical movement and diversified search strategies. Engineering Optimization, pp. 1–21.
[53] Deb, K. (2001). Multi-Objective Optimization Using Evolutionary Algorithms. Systems and Optimization Series, vol. 16. Chichester, U.K.: John Wiley and Sons.
[54] Price, K. V., Storn, R. M. and Lampinen, J. A. (2005). Differential Evolution A Practical Approach to Global Optimization.
[55] Wickramasinghe, U. and Li, X. (2008). Choosing Leaders for Multi-objective PSO Algorithms Using Differential Evolution. In Simulated Evolution and Learning. (pp. 249–258). Springer.
[56] Grobler, J. and Engelbrecht, A. P. (2009). Hybridizing PSO and DE for improved vector evaluated multi-objective optimization. In IEEE Congress on Evolutionary Computation. pp. 1255–1262.
[57] Parsopoulos, K. E., Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P. and Vrahatis, M. N. (2004a). Vector evaluated differential evolution for multiobjective optimization. In Congress on Evolutionary Computation, CEC2004, vol. 1. pp. 204–211 Vol.1.
[58] Li, X. (2004). Better Spread and Convergence: Particle Swarm Multiobjective Optimization Using the Maximin Fitness Function. In Genetic and Evolutionary Computation (GECCO–2004). Springer, pp. 117–128.
[59] Balling, R. (2003). The Maximin Fitness Function; Multi-objective City and Regional Planning. In Evolutionary Multi-Criterion Optimization. Springer, pp. 1–15.