Smart Computing Review

Multi-Objective Particle Swarm Optimization: An Introduction

Vipin Kumar and Sonajharia Minz

School of Computer and Systems Sciences, Jawaharlal Nehru University / New Delhi, INDIA / {rt.vipink, sona.minz}@gmail.com

* Corresponding Author: Vipin Kumar

Received July 20, 2014; Revised September 10, 2014; Accepted September 13, 2014; Published October 31, 2014

Abstract: In the real world, reconciling a choice between multiple conflicting objectives is a common problem. Solutions to a multi-objective problem are those that have the best possible negotiation given the objectives. An evolutionary algorithm called Particle swarm optimization is used to find a solution from the solution space. It is a population-based optimization technique that is effective, efficient, and easy to implement. Changes in the particle swarm optimization technique are required in order to get solutions to a multi-objective optimization problem. Therefore, this paper provides the proper concept of particle swarm optimization and the multi-objective optimization problem in order to build a basic background with which to conduct multi-objective particle swarm optimization. Then, we discuss multi-objective particle swarm optimization concepts. Multi-objective particle swarm optimization techniques and some of the most important future research directions are also included.

Keywords: Multi-objective particle swarm optimization, multi-objective optimization problem, particle swarm optimization, pareto-optimality

Introduction

In the real world, an optimization problem may have more than one objective. The objectives of the problem normally conflict. Therefore, the best compromises between the given objectives generate a set of solutions to the given problem. Particle Swarm Optimization (PSO) [1] is a heuristic-based optimization technique. It is used to consider the social behavior of flocking of birds and fish schooling. PSO is an efficient and simple population-based technique. Therefore, it can be naturally extended to deal with the Multi-Objective Optimization Problem (MOOP).

For MOOP, PSO can be modified in two ways. First, each objective function is treated separately, and second, all objective functions are evaluated for each particle. Generally, a non-dominant solution (best position) is used to guide the particles, called a leader. In each iteration, non-dominant solutions are stored to detect Pareto-optimal solutions that are

DOI: 10.6029/smartcr.2014.05.001

then stored in memory called an external archive. The following fundamental key issues are considered for the design of Multi-Objective Particle Swarm Optimization (MOPSO) such as:

- How to evaluate the objective functions
- How to select the leaders from the archive
- How to promote diversity in the external archive
- How to maintain the archive
- The neighborhood topology used for information exchange

The above issues are discussed in detail in later sections. In this paper, the authors build the basic concepts of PSO and MOOP, and then design the MOPSO. In Section 2, the PSO is described with basic concepts, particle swarm topologies, and parameter selection for the particle swarm. In the Section 3, we describe the concept of dominance, Pareto-optimality, and the procedure to identify a non-dominant set. Section 4 describes the MOPSO concept and its approaches. Finally, Section 5 and 6 have future research directions and our conclusions, respectively.

Particle Swarm Optimization

PSO is the expansion of animal social behavior that follows a population-based meta-heuristic strategy for optimization. It incorporates the acceleration by distance and velocity matching by nearest matching. In the mid- 1990s, it was introduced by J. Kennedy and R. C. Eberhart [1]. Originally, it was utilized to balance the weights in neural networks [2]. The PSO cornerstones can be described as follows:

1. ***Social*** [3]: Human intelligence learns from social interaction. It learns from experiences, which allows it to adapt to the environment and determine optimal behaviors and attitudes. Other fundamental concepts of the social cornerstone indicate that due to mutual learning, individuals become similar, which is called Culture [4].
2. ***Principles of Swarm Intelligence*** [1, 3, 5, 56]: Five fundamental principles can be considered as follows [4]:
    1. *Proximity:* The population should have time computation and simple space.
    2. *Diversity:* The population should have diversity to avoid excessively narrow channels.
    3. *Stability:* An environmental change should not affect the population.
    4. *Quality:* The population should consider quality factors in the environment.
    5. *Adaptability:* The population should be able to change when the computational prize is worth it.

There are many key aspects that have led to PSO becoming so popular:

1. Relatively simple and easy to implement
2. Require fewer parameters to adjust and robust control parameters
3. Effective memory capability to store individual’ s and neighborhood’ s best values
4. Less sensitive to the objective and the parameters
5. Efficient to maintain swarm diversity [7]
6. Very effective in a variety of applications
7. High-quality stable solution with low computational cost [3, 11]

A glossary of common terminology and their definitions follows, for clarity:

- **Swarm:** Population size (number of particles).
- **Particle:** An individual member of the swarm, which is a potential solution to the problem.
- ***pbest (personal best):*** The personal best position achieved by a particle so far.
- ***lbest (local best):*** Position of the best particle member within the neighborhood.
- ***gbest (global best):*** Position of the best particle member from the entire swarm.

- **Leader:** A particle that guides the other particles of the swarm toward the best regions in the search space.
- **Velocity (vector):** The direction in which a particle must move in order to improve its position.
- **Inertia Weight:** The impact of the previous velocity on the current velocity of the particle, denoted by w.
- **Learning Factor:** The attraction of a particle towards either its previous or its neighbor's values. PSO adopts two learning factors, *cognitive learning factor* (c_1) and *social learning factor* (c_2). Cognitive leaning factor represents the attraction toward the particle's own success, and social learning factor represents the attraction toward neighbors. Both factors are constants usually considered in the experiments.
- **Neighborhood Topology:** The set of particles involved to determine the *lbest* value of the given particles.

In PSO, the manipulation of a swarm is different from the evolutionary algorithms, because it promotes a cooperative model rather than a competitive model. An adaptable velocity vector is used by PSO, which changes particle position at each iteration of the algorithm. It exploits information springing from own previous experiences to move toward the promising regions of the search space [9]. To remember previous experience, it has a separate area of memory to store the best position visited in the search space. PSO is described more formally in the context of single-objective optimization as given below.

Let $f: S \to \mathbb{R}$ be the objective function, where S is the d-dimension search space and n is the number of particles, and where $S = \{x_1, x_2, x_3, \dots, x_n\}$. Therefore, the i^{th} particle of the swarm can be represented as $X_i = (x_{i,1}, x_{i,2}, x_{i,3}, \dots, x_{i,d}) \in S$ and the best previous position ever visited by X_i, the search, is as $pbest_i = (pbest_{i,1}, pbest_{i,2}, pbest_{i,3}, \dots, pbest_{i,d}) \in S$. The velocity of the i^{th} particle is $V_i = (v_{i,1}, v_{i,2}, v_{i,3}, \dots, v_{i,d})$. Therefore, the particle movement is computed for the $(t+1)^{th}$ iteration as follows:

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

$$V_i(t+1) = V_i(t) + c_1 r_{i,1}(t) \times \big(pbest_i(t) - X_i(t)\big) + c_2 r_{i,2}(t) \times \big(gbest(t) - X_i(t)\big) \tag{2}$$

where $i = 1,2,3 \dots, n$. The i^{th} particle position and velocity at the t^{th} iteration is denoted as $X_i(t)$ and $V_i(t)$, respectively. At the t^{th} iteration, the best position founded by the entire swarm and the particle itself so far, respectively, are denoted as $gbest(t)$ and $pbest_i(t)$. c_1 and c_2 are the two positive constant acceleration coefficients, which denote cognitive and social parameters, respectively. $r_{i,1}$ and $r_{i,2}$ are two independent randomly distributed values within the range of [0, 1].

To update the velocity, there are three major components [10]:

1. The first component (V_i) models the tendency to continue in the same direction.
2. The second component ($pbest_i$) is a linear attraction toward the personal best position ever found, which is scaled by random weight $c_1 r_{i,1}$.
3. The third component ($gbest$) is a linear attraction towards the global best position found by any particle of the swarm, which is scaled by another random weight $c_2 r_{i,2}$.

The overall procedure of PSO is shown in Table 1.

■ Particle Swarm Topologies

The major components above indicate that the performance of PSO is influenced by the personal best positions ($pbest$) and global best position $gbest$. Therefore, the best positions are heavily dependent on information exchange between neighborhood particles. The particles can be connected to each other in any way. Two general types of neighborhood topologies have been studied for global best ($gbest$) and personal best ($pbest$) [3]. The following are the most common adapted neighborhood topologies:

Empty Graph: Each particle is connected to itself and is compared with its own current position, which is found so far as ($pbest$) [8]. Therefore, in this case Eq. 2 is calculated with $c_2 = 0$.

Local Best: In this topology, the k-immediate neighbors' ($lbest$) performance affects each particle, and each particle is also affected by its own past experience ($pbest$) [8]. In this case, Eq. 2 has $leader = lbest$. If the topology has two immediate neighbors ($k = 2$), then the structure is a *ring topology* and is shown in Figure 1(a).

Fully Connected Graph: In this topology, all members of the given swarm are connected to one another. It can also be observed as the opposite of an *empty graph.* Each particle uses its own best solution ($pbest$)so far, and the best position of

the particle from the whole swarm ($gbest$). This kind of structure is called *star topology* [11]. Therefore, in this case, Eq. 2 is calculated with $leader = gbest$. Star topology is shown in Figure 1(b).

Star Network: In this topology, one particle, called the *focal particle*, is connected to all the remaining particles in the swarm, but each is connected to that one only. In the PSO community, this type of topology is called a *wheel topology* (shown in Figure 1(c)) [8]**.** In this case, $leader = focal$ is used to calculate Eq. 2. A *focal* particle is the only particle that can communicate with all the others, therefore other particles are isolated from one another. The *focal* particle adjusts the other particles' trajectory by comparing the performances of the all particles in the swarm.

Tree Network: All the particles are arranged in a tree structure in which each node has exactly one particle [61]. In the PSO community, this structure is called a *hierarchical* topology (shown in Figure 1(d)). In this case, Eq. 2 is calculated with $leader = pbest_{parent}$.

Ring and wheel are the two most common topologies. Kennedy [12] suggested that a fully connected topology converges very fast, but there is a chance to be trapped in local minima.

Table 1. Overall procedure of the PSO

```
begin
  for each particle of the swarm
    Initialize particles position and velocity randomly
  end for
  do
  for each particle of the swarm
    Evaluate the fitness function
    if the objective fitness value is better than the personal best objective fitness value (pbest) in history
      Current fitness value set as the new personal best (pbest)
    end if
  end for
  From all the particles or neighborhood, choose the particle with best fitness value as the gbest or lbest
  for each particle of the swarm
    Update the particle velocity according to Eq. 2
    Update the particle position according to Eq. 1
  end for
until stopping criteria is not satisfied
end begin
```

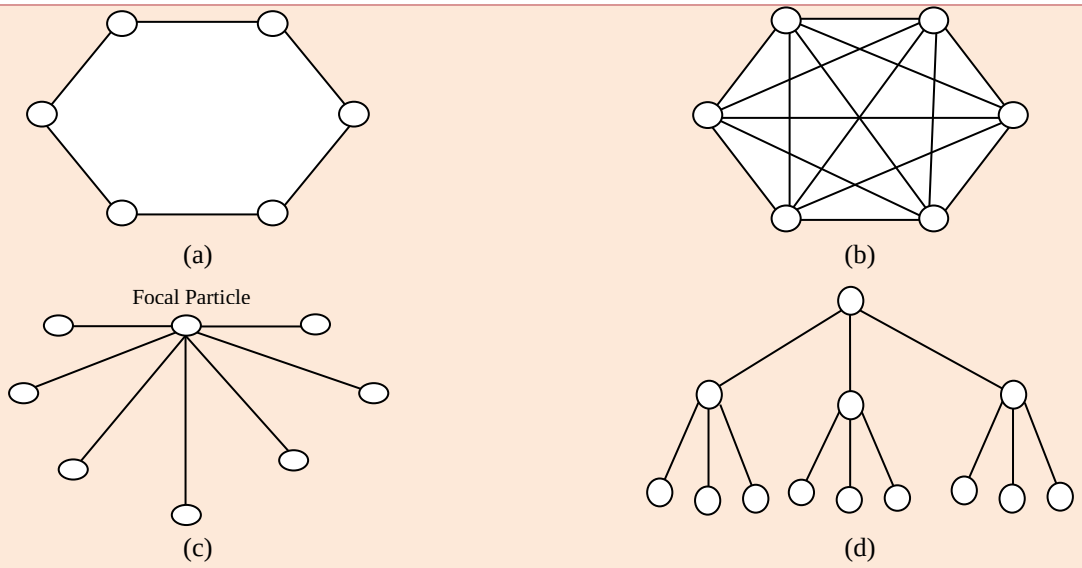

Figure 1. Topologies of the swarm: (a) ring neighbourhood topology (b) fully connected topology (c) star network topology (d) tree network topology.

■ Parameter Selection for Particle Swarm

During the implementation of the PSO algorithm, many considerations are required to facilitate the prevention of swarm explosion and convergence. These considerations include selecting acceleration constants, limiting the maximum velocity and inertia constant, or the constriction factor [4].

a) ***Maximum velocity selection:*** The velocity of the particle is a stochastic variable. Therefore, it creates an uncontrolled trajectory to follow wider cycles in the problem [3, 13]. The upper and lower limits of the velocity are defined to avoid this problem as follows [3]:

$$\textbf{\textit{if}}\ V_{id} > V_{max}\ \textbf{\textit{then}}\ V_{id} = V_{max}$$
$$\textbf{\textit{else if}}\ V_{id} < -V_{max}\ \textbf{\textit{then}}\ V_{id} = -V_{max}$$

If velocity V_{max} is very large, then there is a possibility to move beyond the solution space. On the other hand, if velocity V_{max} is too small, then the movement of the particle is limited. Therefore, an optimal solution may not be obtained. According to the problem characteristics, the value of V_{max} is selected empirically. H. Fan [14] proposed a maximum velocity to ensure that the uniform velocity throughout the all dimensions as shown in Eq. 3.

$$V_{max} = (X_{max} - X_{min})/N \tag{3}$$

X_{max} and X_{min} are the maximum and minimum values of the particle position found so far, and the number of intervals N in the k^{th} dimension that are selected by the users.

b) ***Acceleration constants selection:*** Acceleration of $pbest$ and $gbest$ are controlled by the acceleration constants c_1 and c_2, respectively. The larger values of the acceleration constant may diverge the particles, and small values may limit movement. Ozcan and Mohan [15] carry out several experiments in order to study the effects of a deterministic acceleration constant $c = c_1 + c_2$. The author concluded that the trajectory of the particle goes to infinity when $c > 4$. The high value of the acceleration constant will be limited by V_{max}. Ozcan and Mohan [15] suggested that a good starting acceleration constant point is $c_1 = c_2 = 2$. It is important to understand that c_1 and c_1 may differ according to the problem characteristics.

c) ***Inertia constant and constriction factor selection:*** In the literature, the acceleration constant and maximum velocities are well defined, but the particles may still go to infinity, which is called an explosion of the swarm. Therefore, two methods have been proposed in the literature to control the explosion of the swarm, namely *inertia constant* [16, 17] and *constriction factor* [18, 58].

Inertia constant: The inertia (w) is only multiplied by the velocity $V_i(t)$ at previous time step t. Therefore in Eq. 2, the velocity of the particle can be rewritten as [3]:

$$V_i(t+1) = w(t) \times V_i(t) + c_1 r_{i,1}(t) \times \left(pbest_i(t) - X_i(t)\right) + c_2 r_{i,2}(t) \times \left(gbest(t) - X_i(t)\right) \tag{4}$$

The inertia constant can be a fixed or dynamically changing value [19, 59]. Inertia weight w(t) can be dynamically scaled to the previous velocity that is defined as in Eq. 5:

$$\mathrm{w(t)} = \frac{(\mathrm{T-t}) \times (\mathrm{w_s - w_e})}{\mathrm{T}} \tag{5}$$

where T is the maximum number of time steps to search in the swarm, w_s is the starting inertia weight, and w_e is the ending inertia weight. The starting inertia weight, w_s, is typically 0.9, because it allows for finding a global optimum quickly. To shift from an exploratory nature to an exploitative nature, the ending inertia weight keeps on decreasing until 0.4 [16, 17].

Constriction factor: Clerc and Kennedy [58] proposed a constriction factor to control a swarm explosion. If particles are in multidimensional space, then the velocity Eq. 2 can be rewritten as [3]:

$$V_i(t+1) = \chi \begin{pmatrix} V_i(t) + c_1 r_{i,1}(t) \times \left(pbest_i(t) - X_i(t)\right) \\ + c_2 r_{i,2}(t) \times \left(gbest(t) - X_i(t)\right) \end{pmatrix} \tag{6}$$

where

$$\chi = \frac{2k}{\left|2 - \varphi - \sqrt{(\varphi^2 - 4\varphi)}\right|} \tag{7}$$

where $\varphi = c_1 + c_2,\ \varphi > 4$; typically $k = 1$ and $\varphi = 4.1$ [20]. The value of constriction factor χ is thus 0.729. Therefore, the previous velocity is multipled by 0.729.

Multi-Objective Optimization Problem

Many varieties of real-world problems are available that are concerned with more than two conflicting objectives. These kinds of problems are known as multi-objective problems. The solutions to multi-objective problems are those that have the best possible negotiation among all given objectives [21]. Therefore, multi-objective optimization is required to find the best possible negotiated solutions. To better understand the multi-optimization problem, first we will discuss and define the single objective optimization problem followed by a multi-objective optimization problem (MOOP).

Single-Objective Optimization

Single-objective optimization can be represented for a minimization or maximization problem as:

$$\text{Minimize/Maximize } f(x) \tag{8}$$

subject to:

$$g_i(x) \leq 0, i = 1,2,3,\dots,m \tag{9}$$

$$h_j(x) = 0, i = 1,2,3,\dots,p \tag{10}$$

where $g_i(x) \leq 0$ and $h_j(x) = 0$ are the constraints that satisfy the minimizing or maximizing of $f(x)$, while optimizing and $x \in S \subset \mathbb{R}^n$, where S is a solution space (Definition 1) [22]:

Definition 1 (Single Objective Optimization Problem):

"*A single objective optimization problem is defined as minimizing or maximizing of objective* $f(x)$, *subject to* $g_i(x) \leq 0, i = 1,2,3,\dots,m$ *and* $h_j(x) = 0, i = 1,2,3,\dots,p$; *where* $x \in S \subset \mathbb{R}^n$ *and* S *is the solution space (decision variable). A solution minimizes or maximizes the scalar* $f(x)$, *where n-dimensional decision variable vector* $x = (x_1, x_2, x_3, \dots, x_n)$".

Single objective optimization is to find the many unique global optimums. Single objective global minimum optimization is represented in Definition 2 [22]:

Definition 2 (Single Objective Global Minimum Optimization):

"*Given a function* $f: S \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}, S \neq \emptyset$ *for* $x \in S$ *the value* $f^* \triangleq f(x^*) > -\infty$ *is called a global minimum if and only if* $\forall x \in S: f(x^*) \leq f(x)$. x^* *is by definition the global minimum solution,* f *is the objective function, and the set S is the feasible region of* x . *The global of determining the global minimum solution(s) is called the global optimization problem for the single objective problem.*".

Single-objective optimization has a unique solution, but MOOP may have an uncountable set of solutions that come from trade-offs in objective space.

Multi-Objective Optimization Problem

MOOP minimizes or maximizes the number of objective functions simultaneously. MOOP is also called a *vector optimization*, *multi-performance*, or *multi-criteria* problem [23]:

A general MOOP is represented as follows [24]:

$$Minimize\ /\ Maximize\ f_n(x) \tag{11}$$

where $n = 1,2,3,\dots,N$

Subject to

$$\left.\begin{array}{ll} g_i(x) \geq 0, & i = 1,2,3,\dots,m \\ h_k(x) = 0, & k = 1,2,3,\dots,p \\ x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1,2,3,\dots,q \end{array}\right\} \tag{12}$$

Decision variable: x is a vector of n decision variables which can be written more suitably as:

$$x = [x_1, x_2, x_3, \dots, x_n]^T \tag{13}$$

where T is the transposition of a column vector to a row vector. Decision variables are numerical quantities that are chosen for optimization.

Constraints: Restrictions imposed on the available resources and environment (e.g. time restriction, physical limitation, etc.) are called constraints. They are described as dependence among the parameters and decision variables that are involved in the problem. The last set of constraints, $x_i^{(L)} \leq x_i \leq x_i^{(U)}$, restricts each variable x_i to take a value within a lower $x_i^{(L)}$ and upper $x_i^{(U)}$ bound. These bounds are called the decision space [24]. $g_i(x)$ and $h_k(x)$ constraints are called inequality and equality constraints, respectively. The inequality constraints greater-than-equal-to can be converted into less-than-equal-to and vice versa by multiplying by -1.

- *Infeasible solution:* A solution x that does not satisfy all constraints and variable bounds of the problem is called an infeasible solution.
- *Feasible solution:* A solution x that satisfies all the constraints and variable bounds is known as a feasible solution.
- *Feasible region:* A set of all feasible solutions is called a search space or feasible region, denoted as S.

It is important to have some evaluation criteria to know the goodness of certain solutions. The criteria to express as computable functions of the decision variable are called the *objective function* [22]. The N-number of objective functions can be written more conveniently as:

$$f(x) = [f_1(x), f_2(x), f_3(x), \dots, f_N(x)]^T \tag{14}$$

The multi-objective optimization comprises a multi-dimensional space, called *objective space*. There are two Euclidean spaces which are considered in the MOOP, namely the decision variable and objective space. For each decision variable, there exists a point on objective space, O. It can be represented as:

$$f(x) = [f_1(x), f_2(x), f_3(x), \dots, f_N(x)]^T = O = [O_1, O_2, O_3, \dots, O_N]^T \tag{15}$$

The mapping of n-decision variables and N-objective space is shown in Figure 2.

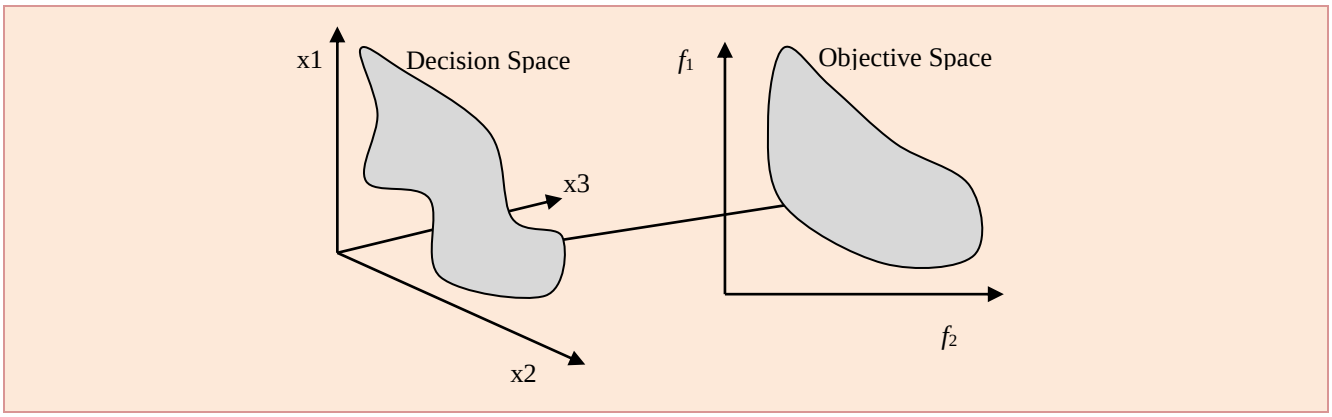

Figure 2. The mapping of decision variable and objective space

MOOP optimizes many objective functions. Therefore, it does not have a unique solution, but a set of solutions. Pareto-optimality Theory [25] is used to find the set of solutions. A mathematical definition of MOOP is defined in Definition 3 [22].

Definition 3 (A General MOOP):

"*Multi-Objective Optimization problem is defined as minimizing/maximizing* $F(x) = (f_1(x), f_2(x), f_3(x), \dots, f_N(x)$ *subject to* $g_i(x) \leq 0,\ i = 1,2,3, \dots, m$ *and* $h_k(x) = 0, k = 1,2,3, \dots, p;\ x \epsilon S$. *An MOOP solution minimizes/maximizes the components of a vector* $F(x)$ *where* $x = (x_1, x_2, x_3, \dots, x_n)$ *is* n*-dimensional decision variable from some universe* S. *It is denoted that* $g_i(x) \leq 0$ *and* $h_k(x) = 0$ *represents the constraints that must be fulfilled while minimizing/maximizing*$vF(x)$, *and* S *contains all possible* x *that can be used to satisfy an evaluation of* $F(x)$."

The objective functions may be continuous or discrete and linear or non-linear in nature. The decision variables can be discrete or continuous.

Linear and Non-linear MOOPs: Linear and Non-linear MOOPs are defined on the basis of objective functions and related constraints. If all objective functions and constraints of the optimization problem are linear, then the MOOP is called a *Multi-Objective Linear Program* (MOLP). However, if any one of the objective functions or constraints is nonlinear, then the MOOP is a *nonlinear multi-objective problem* [24].

Convex and Non-convex MOOP: A convex function is defined in Definition 4 [24].

Definition 4 (Convex Function):

A function $f: \mathbb{R}^n \rightarrow \mathbb{R}$ *is a convex function if for any two pair of solutions* $x_1, x_2 \in \mathbb{R}^n$, *the following condition is true:* $F(\alpha x_1 + (1-\alpha)x_2) \leq \alpha F(x_1) + (1-\alpha)F(x_2);\ \forall\ 0 \leq \alpha \leq 1$."

The convex function properties are given as follows:

1. The actual function value is always underestimated by the linear approximation of $F(x)$ in the interval $[x_1, x_2]$.
2. Hessian matrix $F(x)$ is positive definite for all x.
3. A local minimum of the convex function is always a global minimum.

Now, a convex MOOP can be easily understood, which is defined in Definition 5 [24].

Definition 5 (Convex MOOP):

"*A multi-objective optimization problem is convex if all functions are convex and the feasible region is convex (or all inequality constraints are non-convex and equality constraints are linear).*"

■ Ideal Objective Vector and Dominance Concept

In a multi-objective optimization algorithm (MOOA), the concept of the dominance is used to find the solution from their search space. Therefore, the concept of dominance and terms need to be defined properly.

Ideal Objective Vector

There exists an optimal solution for each N-conflicting objective. The ideal objective vector is constructed by individual optimal objective values, which are defined in Definition 6 [24].

Definition 6 (Ideal Objective Vector):

"*Let* $x_i^o = [x_{i,1}^o, x_{i,2}^o, x_{i,3}^o, \dots, x_{i,n}^o]^T$ *be the vector of variables which minimizes/maximizes the* i^{th} *conflicting objectives* $f_i(x)$, *where* $x_i^o \epsilon\, S$ *such that*

$$\left.\begin{array}{l} minimize/maximize\ f_n(x) \\ Subject\ to \quad x \epsilon\, S \end{array}\right\} \tag{16}$$

It means that $f_i(x_i^o) = optimim_{x \epsilon S}\ f_i(x)$, *if the ideal objective function is denoted as* O^*, *then the ideal vector can be written as follows:*

$$O^* = f^* = [f_1^o, f_2^o, f_3^o, \dots, f_N^o]^T\text{"} \tag{17}$$

where f^* *is the* i^{th} *objective function minimum/maximum.*

Form Eq. 17 above it can be observed that for each objective function, it is not necessary to have the same minimize/maximize solution. Therefore, in general, an ideal vector is non-existing solution. It is only possible when all objective functions are non-conflicting and have same minimum/maximum value to the MOOP. But, the ideal objective vector is used as a reference solution or normalized objective values in a common range in the algorithms. Form Figure 3 it is clear that the solutions nearer to the ideal objective vector are the better solutions.

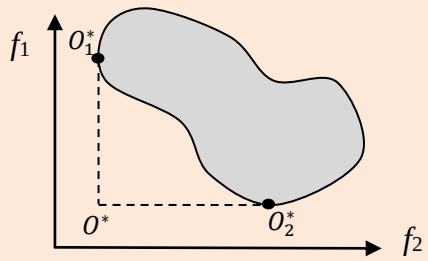

Figure 3. The ideal objective vector

Dominance Concept

The concept of dominance is used in many MOOAs. MOOP produces a set of solutions because of conflicting objective functions. If solution x_i is better than the solution x_j, then solution x_i dominates solution x_j. The definition of domination is given in Definition 7 [24].

Definition 7 (Dominance):

Solution x_i *dominates the other solution* x_j *if the following conditions are true:*

1. x_i *is better than the* x_j *solution for all objectives* ($f_k(x_i) > f_k(x_j)$; $\forall\, k = 1,2,3, \dots, N$).
2. x_i *is strictly better than* x_j *for at least one objective* ($f_{\overline{k}}(x_i) > f_{\overline{k}}(x_j)$; *for at least one* $\overline{k} \epsilon \{1,2,3, \dots, N\}$)

Dominance relations have the following properties, namely not reflexive, not symmetric, not anti-symmetric, transitive and possessive. If solution x_i does not dominate the solution x_j, then it does not mean that x_j dominates x_i, called a possessive property.

■ Pareto-optimality

The comparison of all possible pairs of solutions is performed to get the non-dominant set of solution for a given finite set of solutions. The definition of a non-dominant set is given in Definition 8 [21].

Definition 8 (Non-dominant Set):

"*Vector decision variable* $x \in S \subset \mathbb{R}^n$ *is non-dominant with respect to* S*, if there does not exist another* $x' \in S$ *such that* $f(x') \prec f(x)$".

In another way, we can say that if $P^* \subset S$ is the set of non-dominant solutions that is not dominated by any other member of the solution set S. Then, P^* is called a *Pareto-optimal set*. The formal definition of pareto-optimal and the pareto-optimal set is given in Definitions 9 and 10 [60].

Definition 9 (Pareto-optimal):

"*A vector decision variable* $x^* \in F \subset \mathbb{R}^n$*(F is the feasible region) is Pareto-Optimal if it is non-dominant with respect to* F."

Definition 10 (Pareto-optimal Set):

"*The Pareto-optimal Set* P^* *is defined by* $P^* = \{x \in F | x \; is \; Pareto - optimal\}$."

The pareto-optimal solutions are also called *non-inferior*, *efficient*, or *admissible solutions*. Each member of the entire set P^* is non-dominant. The pareto-front PF^* is defined in Definition 11 [60].

Definition 11 (Pareto Front):

"*For a given multi-objective optimization problem, let Pareto-optimal set* P^* *and objective function* $f(x)$*, the Pareto Front* PF^**is defined as:*

$$PF^* = \{f(x) \in \mathbb{R}^k | x \in P^*\}"$$

Multi-objective optimization also has a local and global pareto-optimal set, like the single-objective optimization local and global optimum solutions. The definitions of locally and globally pareto-optimal sets are defined in Definition 12 [24] and Definition 13 [22].

Definition 12 (Locally Pareto-Optimal Set):

"*A set* P*, there is exists no solution* y *to dominating any other member* x *of the set, where* $x \epsilon P$ *have the neighborhood such that* $\|y - x\| \leq \varepsilon$ *and* $\varepsilon > 0$ *is small positive number, then solution belong to the set* P *constitute local pareto-optimal set*".

Definition 13 (Globally Pareto-optimal Set):

"*A given function* $f: S \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}^k, S \neq \emptyset, k \geq 2$*, for* $x \in S$ *the pareto front* $PF^* \triangleq f(x_i^*) > (-\infty, \dots, \infty)$ *is called the global minimum if and only if*

$$\forall \; x \in S : f(x_i^*) \preccurlyeq f(x)$$

then, x_i^**, i=1,2,3,…,k is called the global pareto-optimal set; where f is the multiple objective function and* S *is the set of feasible regions.*"

The above definition truly indicates that the globally pareto-optimal set is also a locally pareto-optimal set. The Pareto front PF^* can be collectively represented by non-dominant vectors. In the general case, many points (solutions) are computed to get corresponding objective functions $f(x)$. Then, we find the non-dominant solutions to produce the Pareto front. For two objectives, Figure 4 shows the four scenarios of the objective functions, namely *min-min*, *max-min*, *max-max*, and *min-max*, where objectives f_1 and f_2 are the x-axis and y-axis, respectively, and the solid continuous curve shows the pareto-front.

Weak and strict pareto-optimality is defined in Definitions 14 and 15, respectively [22].

Definition 14 (Weak Pareto-optimality):

"*A point* $x^* \in S$ *is a weakly Pareto-optimal if there is no* $x \in S$ *such that* $f_i(x) < f_i(x^*), \; for \; i = 1,2,3, \dots, N$".

Definition 15 (Strict Pareto-optimality):

"*A point* $x^* \in S$ *is a strictly pareto-optimal if there is no* $x \in S, x \neq x^*$ *such that* $f_i(x) < f_i(x^*), for \; i = 1,2,3, \dots, N$".

There are two conditions for pareto-optimality given by Frinz-John and Krush-Kuhn-Tucker, namely the Fritz-John necessary condition and Krush-Kuhn-Tucker sufficient condition. The necessary and sufficient conditions for optimality are defined in Definitions 16 and 17 [24]. It is assumed that the all objective functions and their constraints in Eq. 11 are continuously differentiable.

Definition 16 (Fritz-John Necessary Condition for Pareto-optimality):

"*A necessary condition for* x^* *to be pareto-optimal is that there exist vectors* $\lambda \geq 0$ *and* $u \geq 0$*; where* $\lambda \in \mathbb{R}^L, u \in \mathbb{R}^J$ *and* $\lambda, u \neq 0$*;such that the following conditions are true:*

1. $\sum_{l=1}^{L} \lambda_l \nabla f_l(x^*) - \sum_{j=1}^{J} u_j \nabla g_j(x^*) = 0$, *and*
2. $u_j g_j(x^*) = 0, \forall\, j = 1,2,3,\ldots,J.$"

Definition 17 (Karush-Kuhn-Tucker Sufficient Condition for Pareto-optimality):

"Let objective functions $f_n(x)$ (Eq. 11) be convex and their constraint functions non-convex. Let the objective and constraint functions be continuously differentiable at a feasible solution x^*. A sufficient condition for x^* to be Pareto-optimality is that there exist vectors $\lambda > 0$ and $u \geq 0$; where $\lambda \in \mathbb{R}^L$, and $u \in \mathbb{R}^J$ such that the following equations are true:

1. $\sum_{l=1}^{L} \lambda_l \nabla f_l(x^*) - \sum_{j=1}^{J} u_j \nabla g_j(x^*) = 0$, *and*
2. $u_j g_j(x^*) = 0, \forall\, j = 1,2,3,\ldots,J.$"

For the non-convex objective and constraint functions above, sufficient conditions for pareto-optimality does not hold.

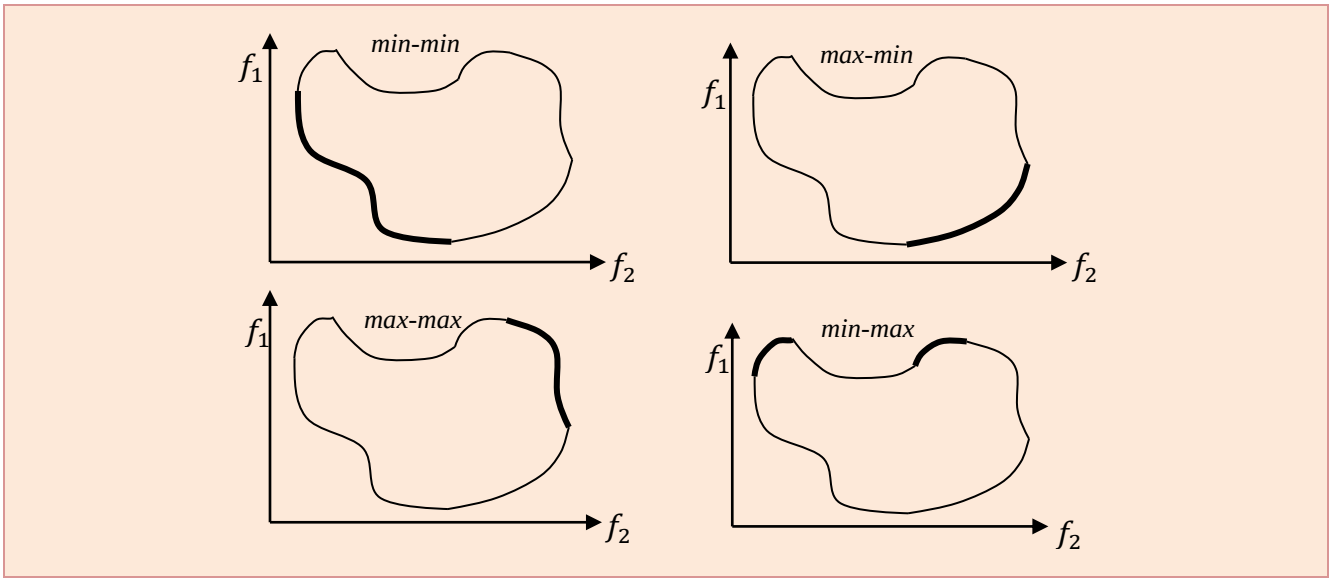

Figure 4. Pareto-optimal solution for the combination of two types of objectives

■ Procedure of Identifying the Non-dominant Set

It is known that the pareto-optimal set has a non-dominant solution set. On the other hand, a non-dominant set may have some pareto-optimal and non-dominant solutions. Therefore, it can be concluded that the non-dominant solution found by an algorithm may not be the ensured true Pareto-optimal set. It is required to find the non-dominant set from the solution space of the problem [24]. In this section, three basic approaches have been discussed, namely, Naive and Slow, Continuously updated, and Kung et al.'s efficient method [26]. In the algorithms, "$x_i \rhd x_j$" indicates that x_i dominates x_j and the number of solutions are N.

Naive and Slow Approach

In this approach, each solution is compared to the remaining solutions of the solution space. The solution is not considered a non-dominant set if a member is dominated by any other member of the remaining solution. Otherwise, the solution is part of the non-dominant solution set. An algorithm for identifying the non-dominant set is shown in Table 2. The complexity of the algorithm is $O(MN^2)$, where N is the number of domination comparisons and M is the number of function value comparisons.

Continuously Updated Approach

In this approach, a partially filled population is checked by every solution from the solution space. This approach is faster than the naïve and slow approach because of better bookkeeping. Initially, it starts from the empty solution set P' and compares all solutions of P' to a solution that belongs to the solution space. If any member of P' is dominated by a solution of the solution space, then that member must be deleted from the set P'. Otherwise, it is ignored. If any solution of P' is not dominant to the solution, then the solution is inserted in the set P'. Stopping criteria is met when all the population of the

solution space is checked. A continuously updated algorithm for identifying the non-dominant set is shown in Table 3. The complexity of this approach is $O(MN^2)$, but the average number of comparisons is smaller than the naive and slow approach.

Table 2. Naive and slow algorithm for identifying the non-dominant set

Begin
Initialization
Non-dominates set $P' = \{\emptyset\}$
Select a solution x_i *from the solution space* S
do
for each solution $x_j \in S$, *where* $i \neq j$,
if *solution* $x_i \rhd x_j$ *then*
$P' = P' \cup \{x_i\}$
end if
$x_i = x_{i+1}$
until $i \leq N$
end

Table 3. Continuously-updated algorithm for identifying the non-dominant set

Begin
Initialization
Non-dominant set $P' = \{x_i\}$, *// x_i is the any member of the solution space*
Select a solution x_j, *where* $i \neq j$ *and* $x_j \in S$ *(solution space)*
Do
if $|P'| == 0$ ***then***
insert one element in set P'
end if
for *a solution* $x_j \in (S - P')$
if $x_j \rhd x_i$ *for any solution* $x_i \in P'$, ***then***
$P' = P' - \{x_i\}$
else
$P' = P' \cup \{x_i\}$
end if
j=j+1
end for
until $i \leq N$
end

Kung et al.'s Efficient Method

In this approach, the population of the solutions are sorted according to the first objective function value. Thereafter, it is divided into top (T) and bottom half (B) sub-populations. It is considered that top first half solution is non-dominant, therefore second bottom half is to be checked for domination with top first half solutions. If the any member of bottom half B is not dominated by the any member of the top half T, then that member is combined with T. This method is most computationally efficient [26].

Multi-Objective Particle Swarm Optimization

Table 4. Kung et al.'s efficient algorithm for identifying non-dominant set

Begin
Initialization
Descending order of importance sorting of the population according to the first objective function values where population is P of size N
if $|P| == 1$ ***then***
return output as P
else
T *=top first half solution of P and*
B *=bottom second half solution of P*
end if
do
for *a non-dominant solution* x_i^B *is not dominated by any* x_j^T *non-dominant solution, where* $x_i^B \in B$ *and* $x_j^T \in T$ ***then***
merge set $\mathcal{M} = T \cup \{x_i^B\}$
$i = i + 1$
until $i \leq |B|$
return $\mathcal{M}$
end

The MOOP has a set of unique solutions (Pareto-optimal Set), therefore it is necessary to modify the PSO to solve MOOP. Eckart Zitzler [27] identified three general goals to archive:

1. Maximum number of solutions in Pareto-optimal set.
2. Minimization of the distance between true Pareto front and the Pareto front produced by an algorithm.
3. Maximum diversity in the solution set found.

There are two methods to find the non-dominant set; first, many runs of PSO where each run of the PSO produces a single solution. Therefore, after several runs of the PSO, a set of solutions is produced. Second, the PSO is a population-based optimization algorithm. Therefore, in a run, it produces a set of non-dominant solutions. Three fundamental issues are considered to design PSO for MOOP as follows [28]:

1. Strategy to choose leaders (particles) for non-dominant solutions to give preference over dominant solution.
2. Strategy to maintain non-dominant set of solutions in the process of searching with respect to all previous and current populations.
3. Strategy to retain diversity in the solution set to avoid convergence to a single solution.

There are two basic PSO approaches for the MOOP [29].

Approach 1: The algorithm considers each objective function separately. One function at a time is evaluated for each particle. The aim of this method is to communicate manipulated information from each objective in order to guide toward a Pareto-optimal front.

Approach 2: All objective functions are evaluated for each particle that considers the concept of pareto-optimality. It produces non-dominant set, called *leaders*.

From the non-dominant set, a leader (a particle) is used to guide particles toward the true pareto-optimal front. Storing the best position of the particles as a Pareto front may exceed the swarm size. The solutions are stored in an *external archive* to avoid the size problem during the search of non-dominant solutions. The external archive is maintained by the replacement of a solution that is dominated by the new solution. Pseudo-code of the general multi-objective particle swarm optimization is shown in Table 5.

Table 5 has some *italics* that makes it different from the general single objective PSO procedure. First, the velocity and position of the particles of the swarm are randomly initialized. Then, a set of leaders are initialized from the swarm with non-dominant particles. For each particle, a leader is selected to update the velocity and position. Some quality measures are also performed for selecting the leader from the archive. Each particle in the swarm $pbest$ is updated, if the previous

$pbest$ is not better than or incomparable with the current one. For each iteration, leaders in the external archive are updated, and the quality of the leader is also recalculated. This process is performed for a fixed number of iterations.

Table 5. Pseudo-code of multi-objective particle swarm optimization

begin
for each particle of the swarm
Initialize particles position and velocity randomly
end for
Initialized external archive (initially empty)
Quality (leader)
do
for each particle of the swarm
select *a particle (leader) from external archive*
Evaluate the fitness function
if the fitness value of the objective is the better than the best fitness value of the objective ($pbest$) in history **then**
Current fitness value of the objective function is set as the new $pbest$
end if
update the velocity of the particle according to the equation-2
update the position of the particle according to the equation-1
end for
update *leader in external archive*
Quality (leader)
until stopping criteria is not satisfied
Report the results of external archive
end begin

Gregorio Toscano Pulido [30] pointed out the algorithmic issues for PSO in a multi-objective scenario, namely, selection and updating of leaders and creation of the new solution.

1. *Selection and updating of the leaders*: How a leader is selected from the non-dominant solution set, where all are incomparable. What should be the addition criteria for leader selection to maintain diversity. What the strategy should be for selecting particles that remain in the external archive in every iteration of the process.
2. *Creation of new solutions*: What should be the criteria to promote the diversity to create the solution in terms of position updating and mutation operator.

It is evident that the selection of a leader is a key concept for MOPSO design. It is easy to consider every non-dominant solution as new leader, and from them one of the leaders is elected. The importance of the leaders is measured by the quality measure criteria that can be define in many ways. Several authors proposed a density-based leader selection measure as the quality measure. In the literature, there are two most important density measures used, namely, the nearest neighbor density estimator [57] and the kernel density estimator [31].

- *Nearest Neighbor Density Estimator:* In this method, the density of a given particle is estimated by the closest neighbor in objective function space.
- *Kernel Density Estimator:* In this method, a parameter is defined, called σ_{share}, that is the radius of the neighbourhood of the particle. The neighborhood is called niche which is defined for each particle. A fewer number of particles is preferred.

One of the complex tasks in MOPSO is to update an external archive. Three archives are used while designing MOPSO. The first is to store the global best solution; the second the personal best value, and the third to store the local best (if required) [60]. If the new solution is non-dominant by all members present in the archive, then the solution is included in the external archive. Moreover, if the new solution dominates some members of the external archive, then the dominant solutions are usually deleted. The bounded archive size is used to avoid the complexity of archive updating [28]. If the archive is full and a new solution is non-dominant by any members of the archive, then diversity is the basic decision

criterion for the insertion of a new solution into the external archive. The archive update is done to retain the maximum diversity of the archive.

MOPSO Approaches

In the literature, many MOPSO approaches have been proposed. Therefore, in this section, some common approaches have been elaborated for simplicity.

1. Weighted objective functions aggregation approach
2. Lexicographic ordering approach
3. Pareto base approach
4. Combined approach

Weighted Objective Function Aggregation Approach

In MOOP, the weighted objective function aggregation approach is the most common approach that is first introduced by Parsopoulos and Vrahatis [32]. In this approach, a weighted sum of the objective is used, shown in Eq. 18:

$$F(x) = \sum_{i=1}^{k} w_i f_i(x) \tag{18}$$

where $i = 1,2,3,\dots,k$ and w_i is a non-negative weight, shown in Eq. 19:

$$\sum_{i=1}^{k} w_i = 1 \tag{19}$$

The weight of the function may be fixed or dynamic during the optimization process. *Conventional Weighted aggregation* (CWA) is used in the case of fixed weights. This method obtains one pareto-optimal solution in each run of the optimization process. Prior knowledge of the search space is required to choose the appropriate weight according to the objective function [33]. Many times a run of the optimization process is required in order to obtain a set of pareto-optimal solutions. This process is not efficient due to the heavy computation of the optimization process and the inability to find solutions in concave regions [34]. There are two methods proposed to avoid the heavy computation, namely *Bang-Bang Weighted Aggregation* (BWA) and *Dynamic Weighted Aggregation* (DWA). In BWA, weights of the bi-objective function can be modified during the process of optimization according to Eqs. 20 and 21:

$$w_1(t) = sign(\sin(\tfrac{2\pi t}{f_w})) \tag{20}$$

$$w_2(t) = 1 - w_1(t) \tag{21}$$

Where f_w and t are the weight change frequency and iteration's index, respectively. The $sign(.)$ function abruptly changes the weight of the objective function. The DWA provides an alternative weight modification that is a slow weight change. The slow weight change tries to keep moving close to the true Pareto-optimal front. The weight is modified according to Eqs. 22 and 23:

$$w_1(t) = |\sin(\tfrac{2\pi t}{f_w})| \tag{22}$$

$$w_2(t) = 1 - w_1(t) \tag{23}$$

The performances of the BWA and DWA are almost equal for the concave pareto front, and BWA is better in the case of the convex pareto front [33].

Lexicographic Ordering Approach

In this method, ranking of the objective function is required by the user according to their importance. Starting from the first ranked objective function, after optimization of the objective functions, the optimal solution x^* is obtained, and the same procedure follows according to the ranking. Let $f_i(x)$ be the i^{th} ranked function where $i = 1,2,3,\dots,n$ indicates the rank of the objective functions. $f_1(x)$ and $f_n(x)$ indicate the most and least important objective functions, respectively [34]. The first objective function is formulated as $Minimize\ f_1(x)$, subject to $g_j(x) \leq 0; j = 1,2,3,\dots,m$ and solution is x_1^* and $f_1^* = (x_1^*)$ is obtained. Likewise, the second objective function can be formulated as $Minimize\ f_2(x)$, subject to $g_j(x) \leq 0; f_1(x) = f_1^*$; where $j = 1,2,3,\dots,m,$ and the solution is x_2^* and $f_2^* = f_2(x_2^*)$ is obtained. These procedures are repeated until the last objective function. Therefore, the problem formulation can be written for the i^{th} objective function as follows [Carlos A. Coello Coello 1999]:

$$Minimize \quad f_i(x) \tag{24}$$

$$\text{subject to} \quad g_j(x) \leq 0; \;\; j = 1,2,3,\dots,m \tag{25}$$

$$f_k(x) = f_k^*; \quad k = 1,2,3,\dots,i-1 \tag{26}$$

x_n^* is the solution of the last ranked objective function that is the desire solution of the problem.

Pareto-based Approaches

Pareto-based approaches use the concepts of the leader selection based on pareto dominance. The leader guides the swarm during a search. As we know that each non-dominant solution is equally good. Therefore, several schemes have been proposed to select a leader (a non-dominant solution) in the literature. The additional criteria is also imposed in order to avoid random searches and fast convergence and the pareto front spread, swarm diversity, and information provided by density estimator. Some of the proposed Pareto-based techniques are categorized based on leader selection techniques in Table 6. It also contains the external archive and neighborhood topology information of the techniques.

Table 6. List of the leader selection techniques in Pareto-based approaches

S.N.	***Leader Selection Techniques***	***External Archive***	***Proposed by*** *(used fully connected neighbourhood topology)*
1.	Dominance	Yes No Yes	Fieldsend and Singh [35] Srinivasan and Hou [36] Alvarez-Benitez et al. [37]
2.	Density Estimator	Yes Yes Yes Yes Yes Yes	Ray and Liew[38] Coello et al. [55] Bartz et al. [39] Li [40] Reyes and Coello [41] Raquel and Naval [42]
3.	Randomly	No Yes	Toscano and Coello[43] Janson and Merkle [44]
4.	Niche Count	Yes Yes Yes	Srinivasan and Hou [36] Li [40] Salazar-Lechuga and Rowe [45]
5.	Sigma Value	Yes	Mostaghim and Teich [46, 47, 48]
6.	Fuzzy Membership	Yes	Zhao and Cao [49]
7.	Stripes	Yes	Vallalabos-Arias et al. [50]

Table 7. List of the leader selection techniques in other approaches

S.N.	***Leader Selection Techniques***	***External Archive***	***Proposed by*** *(used fully connected neighbourhood topology)*
1.	Single Objective	No	Mahfouf et al. [51]
2.	Energy value	Yes	Xiao-hua et al. [52]
3.	Maximum fitness	Yes	Li [53]
4.	Composite leader	No	Zhang et al. [54]

Other Approaches

This subsection concludes those approaches that do not fit in the above categories. Table 7 shows the collection of proposed methods with leader selection techniques, external archives, and neighborhood topology attributes.

Future Research Directions

It can be easily seen that this area is developed during the last decade and has very optimistic expectations. There are many studies still required in this area. Therefore, this section provides some research directions for the future:

- In weighted objective function aggregation, the most fundamental question is to address function weights. Weights of the functions are dependent on the problem at hand. Therefore, there is an extensive investigation of the function weight required according to the characteristics of the problem.
- In the function ordering, the function ranking is performed based on its importance. To rank the objective function, it is required to have prior knowledge about the problem. Therefore, functions ordering according to the function characteristics are the opportunity for future research.
- A fine-tuned MOPSO algorithm with no parameters is a worthy topic of research. It is required to have prior deep knowledge of the relationship of algorithm performance and its parameters with different features.
- In Pareto-based algorithms, there are three crucial issues needed to properly address the selection of leader, diversity promotion, and archive maintenance. Selection of the leader and diversity promotion depend upon swarm dynamics. The promotion of diversity avoids fast convergence and becoming stuck in local optima. It promotes the Pareto solution to move towards the true Pareto front. Therefore, one key research opportunity is to define an efficient diversity promotion method. The archives have a set of leaders that are chosen to the direction of the particle velocity. Therefore, archive maintenance is also a good area of research.

In non-Pareto-based algorithms, information is exchanged among the swarms. Therefore, the direction of information exchange and frequency are needed to investigate properly.

Conclusions

This paper provides an introduction of multi-objective particle swarm optimization, which includes the basic concept of particle swarm optimization and a multi-objective optimization problem. Particle swarm optimization includes the basic key concepts and equations, algorithms, topologies, and parameters descriptions. A multi-objective optimization problem has described the dominance concept, Pareto-optimality and procedure to identify the non-dominant set, which builds the key concept to a better understanding about multi-objective particle swarm optimization. Finally, fundamental issues and the required changes are described to extend particle swarm optimization for a multi-objective optimization problem. The use of external archives, swarm diversity, and leader selection in the swarm, and some common approaches of multi-objective particle swarm optimization are also described. And finally, some of the important future research directions for multi-objective particle swarm optimization are also briefly addressed.

References

[1] J. Kennedy, R. C. Eberhart, "Particle Swarm Optimization," in *Proc. of the IEEE international Conference on Neural Networks*, pp. 1942-1948, Piscataway, New Jersey, USA, 1995.

[2] R. C. Eberhart, R. Dobbins, P. K. Simpson, "Computational intelligence PC Tools," *Morgan Kaufmann Publishers*, 1996.

[3] R. Eberhart, Y. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proc. of the IEEE Congr. Evol. Comput.*, vol. 1, pp. 81-86, 2001.

[4] Y. del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, R. G. Harley, "Particle Swarm Optimization: Basic Concepts, Varients and Applications in Power Systems," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, 2008. Article (CrossRef Link)

[5] S. Yang, M. Wang, L. Jiao, "A quantum particle swarm optimization," in *Proc. of the IEEE Congr. Evol. Comput.*, vol. 1, pp. 320-324, 2004.

[6] M. Millonas, "Swarms, phase transitions, and collective intelligence," in *Proc. of Santa Fe Institute Studies in the Sciences of Complexity*, pp. 417-445, 1994.

[7] A. P. Engelbrecht, "Particle swarm optimization: Where does it belong?," in *Proc. of the IEEE Swarm Intell. Symp.*, pp. 48-54, 2006.

[8] A. P. Engelbrecht, "Computational Intelligence: An Introduction," *John Wiley& Sons*, England, 2002.

[9] K. E. Parsopoulos, M. N. Vrahatis, "Multi-Objective Particles Swarm Optimization Approaches," *IGI global*, Chapter 2, 2008.

[10] D. Boeringer, D. Werner, "Particle swarm optimization versus genetic algorithms for phased array synthesis," IEEE Trans. Antennas Propagat., vol. 52, no. 3, pp: 771-779, 2004. Article (CrossRef Link)

[11] A. P. Engelbrecht, "Fundamentals of Computational Swarm Intelligence," *John Wiley & Sons*, 2005.

[12] J. Kennedy, "Small worlds and mega-minds: Effects of neighbourhood topology on particle swarm performance," in Proc. of IEEE Congr. Evol. Comput., vol. 3, pp. 1931-1938, Jul. 1999.

[13] X. Hu, Y. Shi, R. Eberhart, "Recent advances in particle swarm," in *Proc. of IEEE Congr. Evol. Comput.*, vol. 1, pp. 90-97, 2004.

[14] H. Fan, Y. Shi, "Study on Vmax of particle swarm optimization," in *Proc. of Workshop on Particle Swarm Optimization, Purdue School of Engineering and Technology*, Indianapolis, IN, USA, 2001.

[15] E. Ozcan, C. Mohan, "Particle swarm optimization: Surfing the waves," in *Proc. of IEEE Congress Evol. Comput.*, vol. 3, pp. 1939-1944, 1999.

[16] Y. Shi, R. Eberhart, "Parameter selection in particle swarm optimization," in *Proc. of 7th Int. Conf. Evol. Program*, pp. 591-600, 1998.

[17] Y. Shi, R. Eberhart, "Empirical study of particle swarm optimization," in *Proc. of IEEE Congr. Evol. Comput.*, vol. 3, pp. 1945-1950, 1999.

[18] F. Bergh, A. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, 2004. Article (CrossRef Link)

[19] A. Abido, "Particle swarm optimization for multi-machine power system stabilizer design," in *Proc. of Power Engineering Society Summer Meeting (PES)*, vol. 3, pp. 1346-1351, 2001.

[20] M. Clerc, "The swarm and the queen: Towards a determininistic and adaptive particle swarm optimization," in *Proc. of Congress on Evolutionary Computation (CEC99)*, pp. 1951-1957, 1999.

[21] C. A. C. Coello, "An Introduction to Multi-Objective Particle Swarm Optimizers," *Soft Computing in Industrial Application, Advances in Intelligent and Soft Computing*, vol. 96, pp. 3-12, 2011. Article (CrossRef Link)

[22] C. A. C. Coello, G. B. Lamont, D. A. V. Veldhuizen, "Evolutionary Algorithms for Solving Multi-Objective Problems," *Genetic and Evolutionary Computation Series*, 2nd edition, Springer, 2007.

[23] A. Osyczka, "Multicriteria optimization for engineering design," *Design Optimization*, Academic Press, pp. 193-227, 1985. Article (CrossRef Link)

[24] K. Deb, "Multi-Objective Optimization using Evolutionary Algorithms," *WILEY*, 2002.

[25] M. Ehrgott, "Multicriteria Optimization," *Springer*, Berlin, 2005.

[26] H. T. Kung, F. Luccio, F. P. Preparata, "On Finding the maxima of a set of vectors," *Journal of the Association for Computing Machinary*, vol. 22, no. 4, pp. 469-479, 1975. Article (CrossRef Link)

[27] E. Zitzler, K. Deb, L. Thiele, "Comparison of Multi-objective Evolutionary algorithms: Empirical Results," *Evolutionary Computation*, vol. 8, no. 2, pp.173-195, 2000. Article (CrossRef Link)

[28] C. A. C. Coello, D. A. V. Veldhuizen, G, B. Lamont, "Evolutionary algorithms for Solving Multi-Objective Problems," *Kluwer Academic Publishers*, New York, 2002. Article (CrossRef Link)

[29] M. R. Sierra, C. A. C. Coello, "Improving PSO-based Multi-objective Optimization using crowding, mutation and $\in$-Dominance," *Lecture Notes in Computer Science*, vol. 3410, pp. 505-519, 2005. Article (CrossRef Link)

[30] G. T. Pulido, "On the Use of Self-Adapteation and Elitism for Multiobjective Particle Swarm Optimization," Ph.D Thesis, Computer Science Section, Department of Electrical Engineering, CINVESTAV_IPN, Mexico, 2005.

[31] K. Deb, D, E. Goldberg, "An Investigation of niche abd species formation in genetic Function Optimization," *in Proc. of the Third International Conference on Genetic Algorithms*, pp. 42-50, San Mateo, California, USA, 1989.

[32] K. E. Parsopoulos, M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2-3, pp. 235-306, 2002. Article (CrossRef Link)

[33] Y. Jin, M. Olhofer, B. Sendhoff, "Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does It Work and How?," in *Proc. of GECCO-2001*, 2001.

[34] C. A. C. Coello, "A Comprehensive Survey of Evolutonary-Based Multiobjective optimization Techniques," *Knowledge and Information Systems*, pp. 269-308, 1999. Article (CrossRef Link)

[35] J. E. Fieldsend, S, Singh, "A multiobjective algorithm based upon particle swarm optimisation, an efficient data structure and turbulence," in *Proc. of the 2002 U.K. Workshop on Computational Intelligence*, Birmingham, UK. pp. 37-44, 2002.

[36] D. Srinivasan, T. H. Seow, "Particle swarm inspired evolutionary algorithm (PS-EA) for multiobjective optimization problem," in *Proc. of Congress on Evolutionary Computation (CEC'2003)*, vol. 3, pp. 2292-2297, Canberra, Australia, 2003.

[37] J. E. Alvarez-Benitez, R. M. Everson, J. E. Fieldsend, "A MOPSO algorithm based exclusively on pareto dominance concepts," in *Proc. of 3rd International Conference on Evolutionary Multi-Criterion Optimization*, pp. 459-473, Guanajuato, Mexico, 2005. Article (CrossRef Link)

[38] T. Ray, K. M. Liew, "A swarm metaphor for multiobjective design optimization," *Engineering Optimization*, vol. 34, no. 2, pp. 141-153, 2002. Article (CrossRef Link)

[39] T. Bartz-Beielstein, P. Limbourg, K. E. Parsopoulos, M. N. Vrahatis, J. Mehnen, K. Schmitt, "Particle swarm optimizers for pareto optimization with enhanced archiving techniques," in *Proc. of Congress on Evolutionary Computation (CEC'2003)*, vol. 3, pp. 1780-1787, Canberra, Australia, 2003.

[40] X. Li, A non-dominated sorting particle swarm optimizer for multiobjective optimization," in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO'2003)*, pp. 37-48, 2003. Article (CrossRef Link)

[41] M. R. Sierra, C. A. C. Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and ^{2}-dominance," in *Proc. of 3rd International Conference on Evolutionary MultiCriterion Optimization*, pp 505–519, 2005.

[42] C. R. Raquel, Jr. P. C. Naval, "An effective use of crowding distance in multiobjective particle swarm optimization," in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO2005)*, pp. 257-264, Washington, DC, USA, 2005.

[43] G. T. Pulido, C. A. C. Coello, "Using clustering techniques to improve the performance of a particle swarm optimizer," in Proc. of the Genetic and Evolutionary Computation Conference (GECCO'2004) , pp. 225-237, Seattle, Washington, USA, 2004. Article (CrossRef Link)

[44] S. Janson, D. Merkle, "A new multiobjective particle swarm optimization algorithm using clustering applied to automated docking," in Proc. of Hybrid Metaheuristics, Second International Workshop, pp. 128-142, Barcelona, Spain, 2005.

[45] M. Salazar-Lechuga, J. Rowe, "Particle swarm optimization and fitness sharing to solve multi-objective optimization problems," in *Proc. of Congress on Evolutionary Computation (CEC'2005)*, pp. 1204-1211, Edinburgh, Scotland, UK, 2005. Article (CrossRef Link)

[46] S. Mostaghim, J. Teich, "Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO)," in *Proc. of the 2003 IEEE Swarm Intelligence Symposium*, pp. 26-33, Indianapolis, Indiana, USA, 2003. Article (CrossRef Link)

[47] S. Mostaghim, Jurgen Teich, "The role of dominance in multi objective particle swarm optimization methods," in *Proc. of Congress on Evolutionary Computation (CEC'2003)*, pp. 1764-1771, Canberra, Australia, 2003.

[48] S. Mostaghim, J. Teich, "Covering paretooptimal fronts by subswarms in multi-objective particle swarm optimization," in *Proc. of Congress on Evolutionary Computation (CEC'2004)*, pp. 1404-1411, Portland, Oregon, USA, 2004.

[49] B. Zhao, Y. Cao, "Multiple objective particle swarm optimization technique for economic load dispatch," *Journal of Zhejiang University SCIENCE*, vol. 6, no. A(5), pp.420-427, 2005.

[50] M. A. Villalobos-Arias, G. T. Pulido, C.s A. C. Coello, "A proposal to use stripes to maintain diversity in a multi-objective particle swarm optimizer," in Proc. of the 2005 IEEE Swarm Intelligence Symposium, pp. 22-29, Pasadena, California, USA, 2005.

[51] M. Mahfouf, M.-Y. Chen, D. A. Linkens, "Adaptive weighted particle swarm optimisation for multi-objective optimal design of alloy steels," *Lecture Notes in Computer Science*, vol. 3242, pp. 762-771, 2004. Article (CrossRef Link)

[52] X.-H. Zhang, H.-Y. Meng, L.-C. Jiao, "Intelligent particle swarm optimization in multiobjective optimization," in *Proc. of Congress on Evolutionary Computation (CEC'2005)*, pp. 714–719, Edinburgh, Scotland, UK, 2005. Article (CrossRef Link)

[53] X. Li, "Better spread and convergence: Particle swarm multiobjective optimization using the maximin fitness function," in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO'2004)*, pp. 117-128, Seattle, Washington, USA, 2004. Article (CrossRef Link)

[54] L. B. Zhang, C. G. Zhou, X. H. Liu, Z. Q. Ma, Y. C. Liang, "Solving multi objective optimization problems using particle swarm optimization," in *Proc. of Congress on Evolutionary Computation (CEC'2003)*, pp. 2400-2405, Canberra, Australia, 2003.

[55] C. A. C.o Coello, M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in Proc. of Congress on Evolutionary Computation (CEC'2002), pp. 1051-1056, Piscataway, New Jersey, USA, 2002.

[56] R. Eberhart, J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. of 6th Int. Symp. Micro Machine and Human Science (MHS)*, pp. 39-43, 1995. Article (CrossRef Link)

[57] K. Deb, A. Pratap, S. Agrarwal, T. Meyarivan, "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no .2, pp. 182-197, 2002.

[58] M. Clerc, J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol.Comput.*, vol. 6, no. 1, pp. 58-73, 2002. Article (CrossRef Link)

[59] R. Eberhart, Y. Shi, J. Kennedy, "Swarm Intelligence," *Morgan Kaufmann*, San Mateo, CA, 2001.

[60] M. R. Seirra, C. A. C. Coello, "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, 2006.

[61] S. Janson, M. Middendorf, "A hierarchical particle Swarm Optimizer," in *Proc. of Congress on Evolutionary Computation (CEC' 2003)*, pp. 770-776, Camberra, Australia, 2003.