Chapter 2
Particle Swarm Optimization and Its Applications in the Manufacturing Industry

Pinkey Chauhan and Shashi Barak

1 Introduction to Optimization

The term "optimization" entails the process of optimizing a given mathematical function or system's desirable properties while minimizing its undesirable characteristics. In the most basic sense, the optimization process tries to determine the best possible set of values to attain a given objective by satisfying various restrictions called constraints.

If we consider only one objective, then the problem is mathematically formulated as follows:

Minimize (or maximize)

$$f(x); \quad x = (x_1, x_2, \ldots., x_D) \tag{2.1}$$

subject to, usually defined by

$$F = \left\{ x \in {}^{D} \mid \; h_i(x) = 0; \text{and } g_j(x) \geq or \leq 0 \right\}$$

$$i = 1,2, \ldots.,m \text{ and } j = m+1, m+2, \ldots.,p$$

where $f, h_1, h_2, \ldots, h_m, g_{m+1}, g_{m+2}, \ldots g_p$ are real valued functions defined on $\mathfrak{R}^D$. The function $f(x)$ that is to be optimized (maximized or minimized) is called the "objective function." The equations $h_i(x) = 0$ for $i = 1, 2, \ldots, m$ are known as the equality constraints, and the inequalities $g_j(x) \geq or \leq 0$ for $j = m+1, m+2, \ldots, p$ are called inequality constraints. The independent variables $x_i's$ are called decision variables. A decision vector $x = (x_1, x_2, \ldots, x_D) \in \mathfrak{R}^D$ satisfying all the constraints is

P. Chauhan (✉) · S. Barak
Jaypee Institute of Information Technology, Noida, India

D. Singh et al. (eds.), *Design and Applications of Nature Inspired Optimization*, Women in Engineering and Science, https://doi.org/10.1007/978-3-031-17929-7_2

called a "feasible point or solution." A feasible optimal solution is a possible solution that optimizes the objective function. It is intended to identify the independent variable values x_1, x_2, . . ., x_D that optimize the objective function $f(x)$ without violating any of the constraints specified in problem (2.1)

The problem is known as a "linear programming problem (LPP)" when all of the functions $f(x)$, $h_i(x)$, $g_j(x)$ in the optimization problem are linear. The problem is known as a "nonlinear optimization problem" or a "nonlinear programming problem (NLPP)" if one or more of these functions are nonlinear. The model is termed an "integer programming problem" if the solution adds an extra constraint that the decision variables must be integers. The problem is known as a "mixed integer programming problem" when some of the variables are integers and others are real.

Local and Global Optimal Solution

The solution of an optimization problem is classified by the quality of the solution. The two types of solutions are referred to as local optima and global optima. An optimum $\overline{x}$ (local or global) is defined as follows: Let x be a solution vector of a given optimization problem that which satisfies all constraints. Now, let F be a set of all such solution vectors x, called feasible/solution space. Then, for a minimization problem, if for $\overline{x} \in F$, there exists an ε-neighborhood $N_\varepsilon(\overline{x})$ around $\overline{x}$ such that $f(x) \geq f(\overline{x})$ for each $x \in F \cap N_\varepsilon(\overline{x})$ and then $\overline{x}$ is called a "local minimum" of the given optimization problem. The functional value $f(\overline{x})$ will be called the local minimum value. If, however, $\overline{x} \in F$ and $f(x) \geq f(\overline{x})$ for all $x \in F$, then $\overline{x}$ is called a "global minimum" of the given optimization problem. The functional value $f(\overline{x})$ will be called the global minimum value. The local and global optima of a function are shown in Fig. 2.1.

If the problem is linear in nature, then the local solution will also play the role of a global optimum solution. The local optimum solution for an NLPP is guaranteed to be the global optimal solution, if the objective function for a minimization situation is convex and the domain of definition specified by the set of constraints is also convex. Figure 2.1 illustrates an example of a function with local and global optima. Figure 2.2 shows a function having a unique minimum (an example of a unimodal function), while Fig. 2.3 shows an example of a function having several local and global optima (multimodal function).

Algorithms that aim at determining the global solution are called global optimization algorithms. The practical necessity of global optima in real-life scenarios has motivated researchers to develop several global search methods for solving NLPP efficiently. Global search algorithms are categorized into two types: deterministic and probabilistic techniques. For exhaustively searching the solution space, deterministic approaches rely on a predetermined set of rules. Moreover, the solution found by a deterministic method always depends on the starting conditions and often be suboptimal. Probabilistic methods follow a stochastic approach to search the

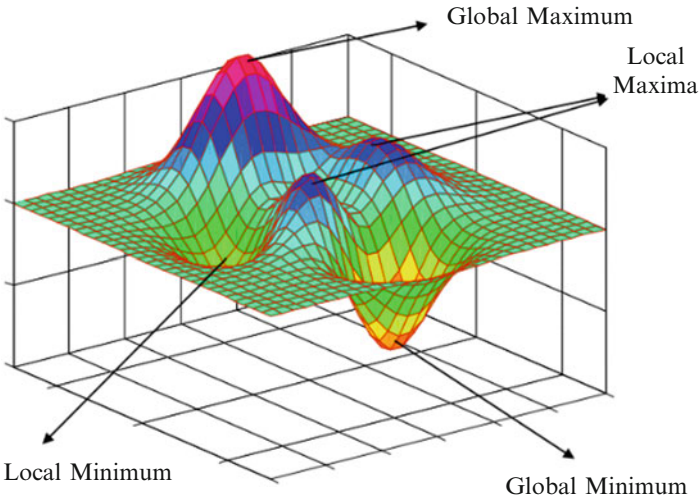


Fig. 2.1 Local optimum and global optimum

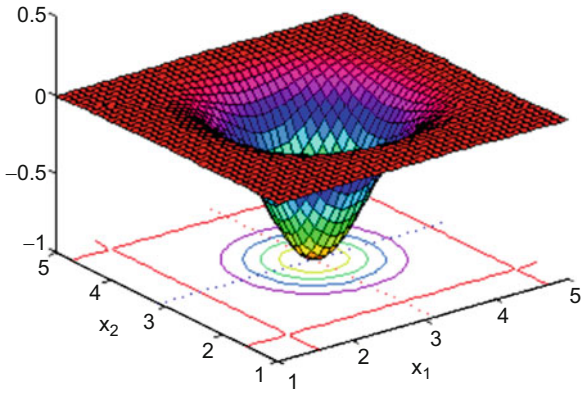


Fig. 2.2 Visualization of a unimodal function

feasible space thoroughly for locating global optimal solutions. The deterministic techniques are applicable to a specific range of functions, such as differentiable or Lipschitz continuous functions, but Stochastic methods are applicable to a much broader range of functions. Despite the fact that probabilistic approaches do not guarantee global optima, they are occasionally recommended over deterministic methods due to their applicability to a broader class of functions. A detailed study of deterministic and stochastic methods could be found in [1–6]. A taxonomy of some global optimization methods is shown in Fig. 2.4

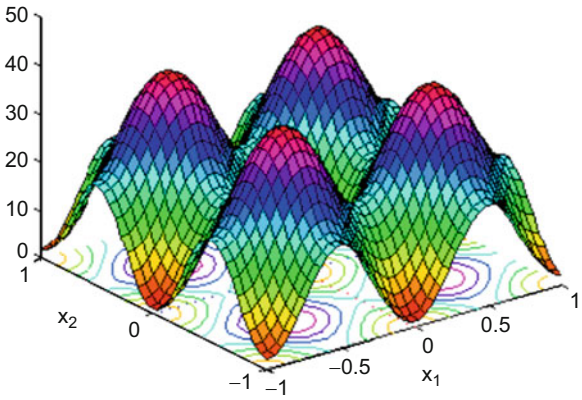


Fig. 2.3 Visualization of a multimodal function

Nature-Inspired Algorithms

Nature-inspired computing is a new computer paradigm that is based on self-organization and complex systems principles. Techniques that simulate an existing natural process to discover an optimum solution to a problem that seems to be resistant to conventional methods are known as nature-inspired optimization algorithms. The behavior or working of biological systems have been inspiring meta-heuristic search algorithms since its inception, for example, genetic algorithms [7], ant colony optimization [8, 9], tabu search [10], bacterial foraging optimization algorithm (BFOA) [11], differential evolution [12], central force optimization [13, 14], artificial bee colony optimization [15], glowworm swarm optimization [16], and particle swarm optimization [17]. The abovementioned methods have the advantage of being able to successfully address a variety of standard or application-based problems without any prior knowledge of the problem space. Furthermore, these algorithms are more capable of finding a problem's global optima. The scope of this chapter is limited to particle swarm optimization (PSO), which is considered an efficient, simple, and popular nature-inspired optimization approach. PSO is a swarm intelligence method, which is inspired by the behavior of fish schools and bird flocks for solving global optimization problems. The next section will present a detailed description of the simulation and parameters of PSO.

2 Particle Swarm Optimization

Particle swarm optimization (PSO) belongs to the category of swarm intelligence techniques, inspired from the well-informed social behavior of organisms. The foraging process of swarm analogies, such as bird flocks and fish schools, is simulated by PSO. This concept was firstly proposed as an efficient heuristic technique by Kennedy and Eberhart in 1995 [17]. The benefits of using PSO include

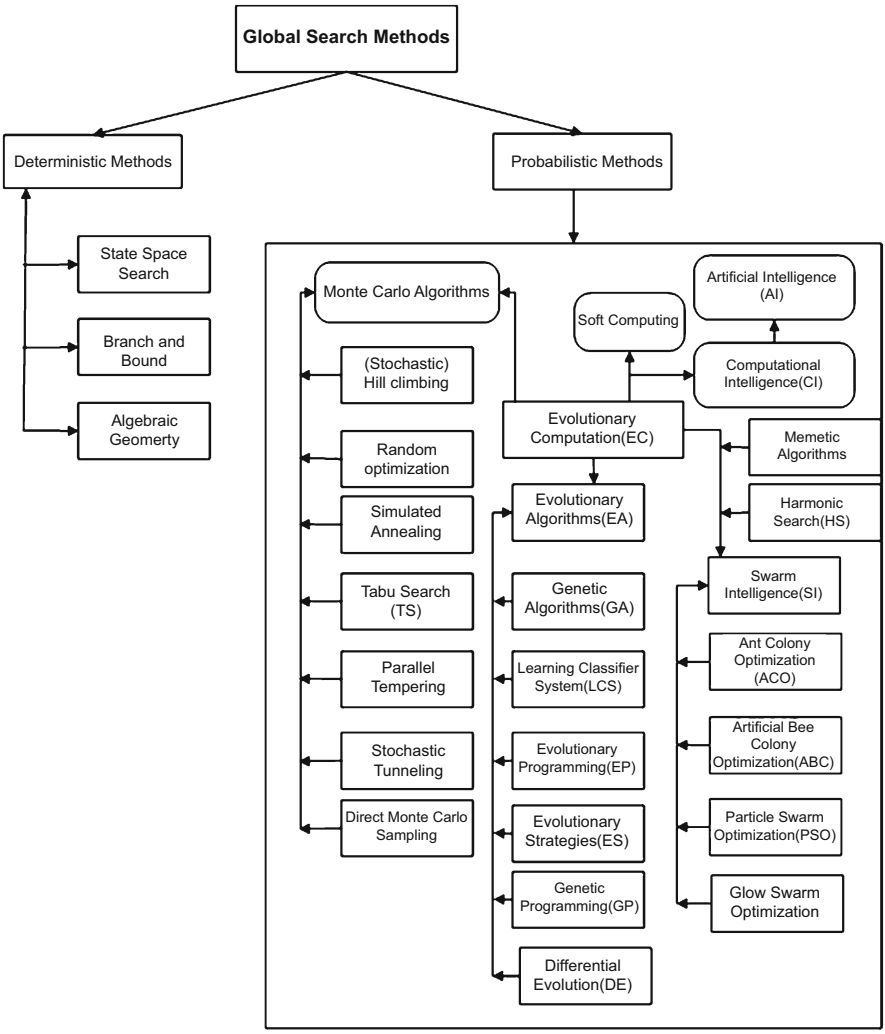


Fig. 2.4 Taxonomy of global optimization methods

fast convergence to the global optimum, a simple to implement code, as well as a complex computation-free environment. The searching process in PSO has better global searching capability at the start of the run and good local searching capability near the end. PSO is an efficient global optimizer that has gained great attention from academics since its inception. Because it is an efficient global optimizer, it may be viewed as an alternative to genetic algorithms (GA) and other evolutionary algorithms (EAs). PSO is an excellent option for dealing with a wide range of problems appearing in biology, economics, engineering, industry, and other real-world domains due to its simple and effective searching technique.

How PSO Works

For a D-dimensional search space, the ith particle of the swarm at time step t is represented by a D-dimensional vector, $x_i^t = \left(x_{i1}^t, x_{i2}^t, \ldots, x_{iD}^t\right)^T$. The velocity of this particle at time step t is represented by another D-dimensional vector, $v_i^t = \left(v_{i1}^t, v_{i2}^t, \ldots, v_{iD}^t\right)^T$. The previously best visited position of the *i*th particle at time step *t* is denoted as $p_i^t = \left(p_{i1}^t, p_{i2}^t, \ldots, p_{iD}^t\right)^T$. This is also called the personal best position or p_{best}.

The velocity of the *i*th particle is updated using the velocity update equation, given by

$$v_{id}^{t+1} = w^* v_{id}^t + c_1 r_1 \left(p_{id}^t - x_{id}^t\right) + c_2 r_2 \left(p_{gd}^t - x_{id}^t\right) \tag{2.2}$$

Here "g" is the index of the best particle in the swarm, and Pgd represents the best particle, i.e., the particle having the best fitness value. This is also called gbest, i.e., the global best.

The position updating rule is given below

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \tag{2.3}$$

where $d = 1, 2\ldots,D$ represents the dimension and $i = 1, 2,\ldots,S$ represents the particle index. S is the size of the swarm, and $c1$ and $c2$ are called cognitive and social acceleration constants, respectively, and constitute the parameters that have to be fine-tuned for the PSO to achieve convergence. $r1$ and $r2$ are uniform random numbers in the range [0, 1] and used to randomize the acceleration constants. Due to the stochastic effect introduced by these numbers, PSO trajectories should be considered stochastic processes. Equations (2.2) and (2.3) define the classical version of PSO algorithm with inertia weight (w).

In the velocity update Eq. (2.2), the new velocity v_{id}^{t+1} can be seen as the sum of three terms:

(i) *Momentum:* The first term $w^* v_{id}^t$ is momentum, which functions as memorization of the particle's prior flight direction. This concept restricts the particle from altering its path abruptly.

(ii) *Cognitive Component:* The second term $c_1 r_1 \left(p_{id}^t - x_{id}^t\right)$, related to local search, is proportional to the vector $\left(p_{id}^t - x_{id}^t\right)$ and leads back particle to its own best position. This factor, also known as the cognitive component of the velocity update equation, controls the step size in the direction of the particle's personal best position.

(iii) *Social Component:* The third term $c_2 r_2 \left(p_{gd}^t - x_{id}^t\right)$ is called social component, which is linked to the global search. This term is proportional to $\left(p_{gd}^t - x_{id}^t\right)$

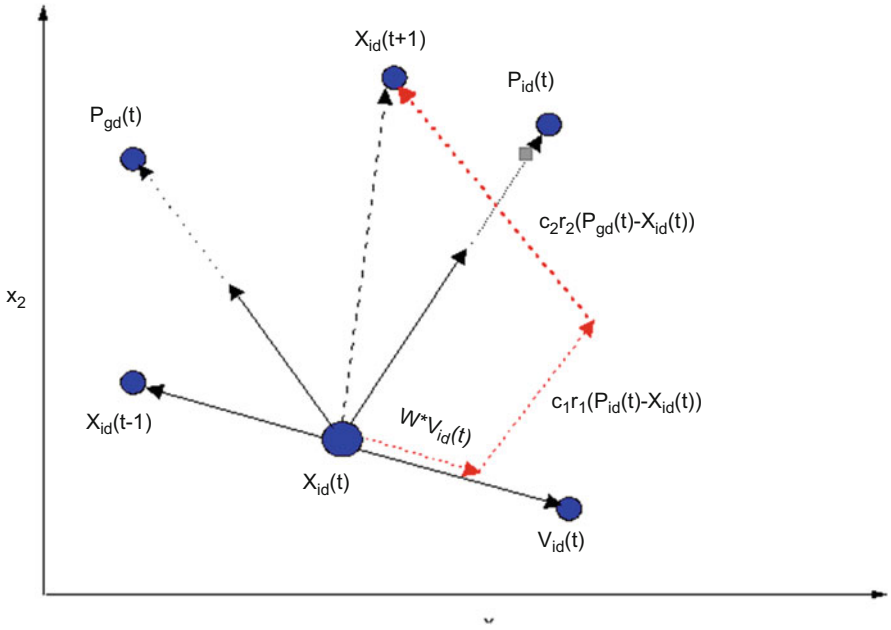


Fig. 2.5 Geometrical visualization of particle's movement in two-dimensional space

and points to the best position in the neighborhood. This term regulates maximum step size in the direction of global best particle.

In order to improve the resolution of the search, a constant, Vmax, is introduced in Eberhart et al. [18] to clamp the velocities of the particles in the range [−Vmax, Vmax]. The maximum velocity, Vmax, acts as a parameter to restrict the global exploration ability of a particle. The movement of a particle in two-dimensional space can be visualized geometrically in Fig. 2.5.

The PSO paradigm follows the five basic principles of swarm intelligence [17, 19].

- Proximity principle: The proximity principle states that the population should be able to do simple spatial and time-related calculations.
- Quality principle: The population should be able to adapt to environmental quality variables.
- Principle of diverse response: The population should not commit its activities along excessively narrow channels.
- Stability principle: The population's behavior should not alter in response to changes in the environment.
- Principle of adaptability: The population must be able to adjust its behavior mode in response to the computational price.

The PSO's searching process is elaborated by the algorithm given below:

Algorithm: Basic PSO

Create and Initialize a D-dimensional swarm, S
For t= 1 to the maximum bound on the number of iterations,
For i=1 to S,
For d=1 to D,
Apply the velocity update equation (2)
Update Position using equation (3)
End- for-d;
Compute fitness of updated position;
If needed, update historical information for P_i and P_g;
End-for-i;
Terminate if P_g meets problem requirements;
End-for-t;

Understanding PSO Parameters

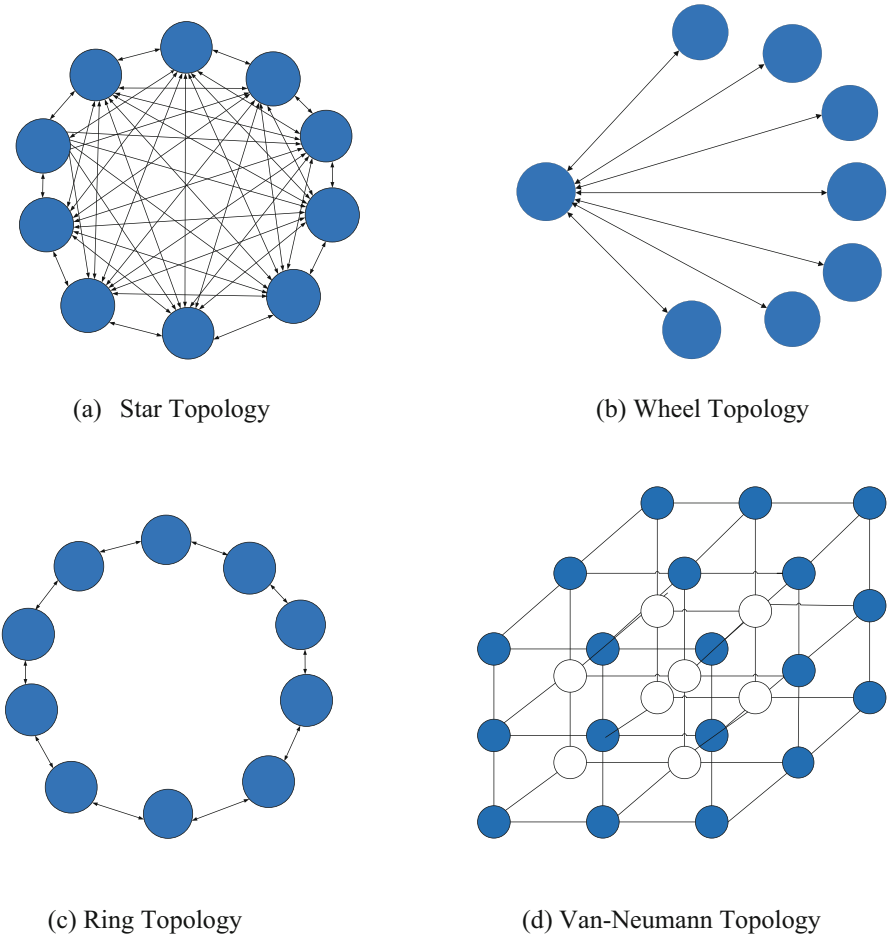
The basic PSO has a number of parameters that should be fine-tuned to regulate the performance of the algorithm in a desired way. These parameters are briefly defined as follows [18, 20].

(i) *Swarm Size:* It refers to the number of random solutions generated initially to start the searching process. A good and diversified initial swarm leads the search in a better way, which may affect the performance of PSO significantly. The swarm size is problem dependent.

(ii) *Acceleration Coefficients.* These parameters are designated by $c1$ and $c2$, and they measure the stochastic impact of a particle's personal and social experiences on total velocity per iteration. With particle speed growing without control, the influence of these settings can make the PSO more or less "responsive" and possibly even unstable. Usually, $c1$ and $c2$ are taken as the following: $c1 = c2 = 2.0$; $c1 = 1.3$, $c2 = 2.8$ and $c1 = 2.8$, $c2 = 1.3$.

(iii) *Velocity Clamping (Vmax).* The concept of velocity clamping was introduced to limit velocities to the range [−Vmax, +Vmax] for each component of v_{id}. The value of parameter Vmax is carefully chosen, because it has a significant impact on the exploration-exploitation trade-off. Vmax's ideal value is problem-specific, and there is no fair rule of thumb.

(iv) *Inertia Weight (w):* Shi and Eberhart [21] introduced it as an explicit parameter to alter the momentum of a particle to a certain extent. The inertia weight governs the contribution of the previous velocity so that particles do not change their directions drastically and head toward good regions. When $w>1.0$, the particle will accelerate to its maximum velocity Vmax (or −Vmax) and then jump off the feasible space, while a value $w<1.0$ will force the particles to slow down until its velocity drops to zero, resulting in localized stagnation. Therefore, the value of w in the range [0.4, 1.0] is preferred more often.



(a) Star Topology (b) Wheel Topology

(c) Ring Topology (d) Van-Neumann Topology

Fig. 2.6 Social networks for particle-to-particle interaction

(v) Particle's Social Interaction: The interaction methods show how the particles are interconnected with each other for information exchange. Some common structures are given below and are illustrated in Fig. 2.6.

- Star Topology: The star social structure is shown in Fig. 2.6, in which all particles are linked to each other and hence the communication occurs within the entire swarm. In this situation, every particle is drawn to the optimal solution traced by the entire swarm. As a result, each particle mimics the overall ideal solution. The "gbest PSO" is the initial version of the PSO, which utilized a star network structure. It has been observed that the gbest PSO converges more quickly than other communication networks but is more prone to becoming stuck in local minima. The "gbest PSO" shows better performance for problems having single optima.

- Ring or Circle Topology: The ring social structure is shown in Fig. 2.6c. The communication between each particle and its closest neighbors occurs within the ring social structure. By advancing toward the neighborhood's best solution, each particle makes an effort to emulate its best neighbor. Due to the link between limited number of particles, the convergence occurs more slowly as compared to star structure, but a bigger portion of the search space is covered in this structure. The above quality of ring social structure recommends it for multimodal problems. The first implementation of PSO using ring structure was named "lbest PSO."
- Wheel Topology: The wheel social structure isolates individuals living in a neighborhood from one another. As shown in Fig. 2.6b, one particle serves as the focal point, via which all information is transmitted. The focus particle evaluates all of the neighbors' performances and shifts its position toward the best neighbor. If the focal particle's changed position leads to improved performance, then the entire neighborhood is informed about the improvement.
- Von Neumann or Square Topology: The von Neumann social structure, as shown in Fig. 2.6d, has particles connected in a grid pattern.

There is no single structure that works best for all problems. It has been observed [20] that ring topology performs better for unimodal problems and star topology provides better results for multimodal problems. Particle indices are commonly used to define neighborhood size.

Binary Particle Swarm Optimization

PSO was originally developed for continuous optimization problems, but it has now been expanded to discrete and binary-valued problem spaces. Kennedy and Eberhart [22] created the first discrete version of PSO for binary issues as a result of their initiative. The core particle searching mechanism is the same in binary PSO as in the continuous version, with the exception of a change in the position update equation, which in the case of binary PSO becomes a binary number generator. To determine whether x_{id}, the d^{th} component of x_i, should be evaluated as "0" or "1," the velocity is employed as a probability threshold. A mapping rule, from vid to a probability in the range [0, 1] must be defined for each $v_{id} \in \Re$. This is accomplished by squashing velocities into a range of [0, 1] using a function called "sigmoid function." The sigmoid function is defined by the following mathematical equation:

$$sigm(v_{id}) = \frac{1}{1 + \exp(-v_{id})} \tag{2.4}$$

The shape of the sigmoid function resembles the shape of the letter "S" as shown below in Fig. 2.7. The sigmoid function trajectory serves the purpose of a probability generating function for deciding a bit (from 0 to 1 and vice versa).

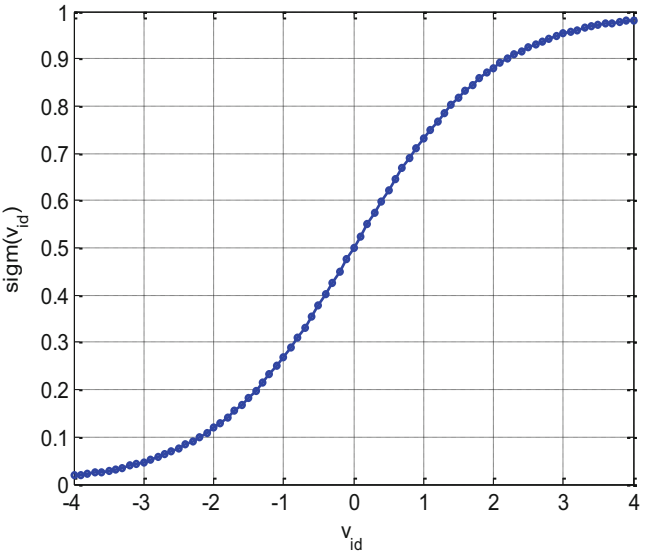


Fig. 2.7 Visualization of sigmoid function

In BPSO, the particle velocities (v_{id}) are fed as input to the sigmoid function, which normalizes them to be produced in the range [0, 1]. The generated values are further employed as the probability threshold for selecting bits 0 or 1. The binary PSO (BPSO) position update equation is now a probabilistic update equation:

$$x_{id}^{t+1} = \begin{cases} 1 & \text{if } U(0,\ 1) < sigm\left(v_{id}^{t}\right) \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

where is a quasi-random number generated from a uniform distribution with values between 0 and 1. If sigm (vid) = 0, xid will remain 0 as shown in Eq. (2.5) (for convenience, the time scripts are dropped). This occurs when either $v_{id} < -10$ or vid $> {}_{10}$ [23]. To overcome this situation, it has been advised to set v_{id} in the range $[-4, 4]$ and to use velocity clamping with Vmax = 4. set vid and employ velocity clamping with Vmax = 4.

Research Developments in PSO

PSO, like several other population-based methods, faces problems while dealing with a certain class of problems, e.g., multimodal and complex real-world problems having a large number of decision variables. Two common drawbacks/shortcomings detected are as follows: premature convergence, when the algorithm converges to a solution which is not optimum; stagnation, when the algorithm shows no

improvement in the fitness value although new particles are generated. These issues arise when the swarm enters a suboptimal state in which the algorithm is no longer capable of producing solutions that improve to the desired accuracy. The main reason behind the occurrence of these problems is the loss of diversity which in turn arises due to an imbalance between exploration (exploring different parts of the search space in order to find a good optimum) and exploitation (ability to narrow down a search to a feasible region in order to fine-tune a potential solution). The focus of this thesis is to develop improved PSO variants so that they can be applied to different problems arising in process industries.

According to the literature, the modification strategies for developing improved PSO variants may be broadly classified as:

(i) Hybridizing PSO with ideas borrowed from other heuristics or traditional methods
(ii) Disturbing the searching process of PSO by introducing some dynamics, e.g., chaotic maps
(iii) Proposing new strategies for parameter selection in PSO

A brief introduction on the developments of PSO using different strategies is presented below:

Mutation Embedded PSO Variants The idea of mutation was originally suggested for genetic algorithms to create perturbation in the population. The work of a mutation operator is to perturb the individuals so as to increase the diversity of the population and to pull out the particles, which are probably stuck in some local optimizer. Many mutation operators have been implemented in PSO so far, including Gaussian, Cauchy, Uniform, Levy, Power, and others [24, 25], to improve its performance and the notion reportedly provided satisfactory results.

Inertia Weight-Based PSO Variants In order to improve PSO's performance, Shi and Eberhart [26] added a new parameter called "inertia weight" to the original PSO, which was designed to manage the swarm's exploration and exploitation abilities by weighting particle motion. In addition, Shi and Eberhart [26] empirically analyzed the effects of inertial weight and maximum velocity on PSO performance, taking into account various parameter settings. Early studies suggested a constant inertial weight throughout the search, whereas later research focused on dynamic changes in inertial weight that dynamically regulated search capabilities. The various inertial weight approaches can be categorized as follows:

Linear Strategy: During a run, an inertia weight that decreases linearly from a reasonably large value to a small value has more global searching capacity at the start and more local searching ability near the finish. Several investigations [27–29] have documented the linear method to choose inertia weight, which depends on time.

Nonlinear Strategy: A nonlinear strategy that changes dynamically over time or iterations, based on the performance of a swarm or particle, was tested in several

experiments and shown to be superior to the linear strategy. This approach gave better results with fewer iterations [30, 31].

Exponential Strategy: Exponential functions, which are faster and decreasing than linear and nonlinear functions, have attracted a lot of attention as a possible alternative for a lowering inertia weight strategy [32, 33]. The findings of the experiments reveal that exponential techniques converged faster than linear strategies early in the search process and produced better solutions.

Adaptive or Self-Adaptive Strategies: Choosing an inertia weight that adjusts to the needs of the particle is seen to be a preferable alternative, and researchers have presented a number of adaptive and self-adaptive ways for choosing an inertia weight that considerably improves PSO performance [32, 34, 35].

Fuzzy Rules-Based Strategy: A fuzzy system-based technique for dynamically adjusting inertia weight as developed by [36]. The input variables are the current best performance evaluation and the current inertia weight, whereas the output variable is the change in inertia weight. [37] developed another fuzzy-based technique in which the inertia weight is dynamically changed using fuzzy sets and rules.

Distribution-Based Random Adjustments: Some implementations used tactics based on probability distribution functions, which were found to be beneficial on a number of levels. Pant et al. [38] proposed a new Gaussian-based inertia weight based on the absolute value of half of the Gaussian random number, as well as discussing the likelihood of utilizing Gaussian and exponential distributions for producing the initial swarm. When the algorithm is likely to be stuck in local optima, Zhu et al. [39] developed a random adjustment for determining inertia weight with an adaptive initialization technique. The modified version was then used to solve the path planning problem for UAVs (unmanned aerial vehicles) and produced effective results.

Chaotic Inertia Weight Strategies: Some methods [40, 41] took advantage of dynamic systems to determine an adaptive inertia weight that would improve swarm diversity and convergence speed of method. The strategies incorporated chaotic terms as an additional parameter to increase randomness and, as a result, population diversity.

Some review reports have also been published by researchers [32, 42, 43] to analyze various existing inertia weight strategies and their performances. These review studies are always been very helpful to researchers, when selecting an existing strategy or proposing a new one.

Chaotic PSO Variants Chaos is a bounded unstable dynamic phenomenon in nonlinear systems that is sensitive to initial conditions and comprises infinite unstable periodic motions. It occurs in a deterministic nonlinear system under deterministic conditions, despite the fact that it appears to be stochastic. Chaos was introduced by many researchers as a disturbance term to enhance the capability of PSO for finding global optima. Chaos was added for handling premature convergence [44, 45], parameter adaptation [46], enhancing exploitation, maintaining population diversity [47], and preventing stagnation phenomena [48]. Many

researchers [49–51] have also developed application-based variants of PSO by adding chaos at suitable points.

Binary PSO Variants Binary PSO was introduced by Kennedy and Eberhart [22] for handling binary and mixed integer problems. Binary PSO has the same searching process as its continuous version, therefore, having issues of premature convergence, and stagnation as well. To overcome the above shortcomings in binary PSO, researchers had come up with new modifications intending to improve the performance of binary PSO. Basic binary PSO uses sigmoid function for generating binary numbers. Some researchers have employed other functions as linear probability function [52], Boolean function [53], and bit change mutation [54, 55]. The sigmoid function is substituted by the Gompertz function in the study by [56], which has characteristics of both sigmoid and linear functions. The computational results shows that the novel approach is efficient over binary PSO and turns out as an efficient and handy algorithm for solving binary-valued problems.

Binary PSO has been extended in a number of studies [57–59] to solve integer or combinatorial optimization problems that arise in a variety of sectors, including science, engineering, and industry.

3 Application of PSO Manufacturing Industry

Large-scale, high-dimensional, nonlinear, and extremely unpredictable nature are all characteristics of industrial problems. Complex optimization problems are frequently solved using traditional methods such as "trial and error." Due to intrinsic limitations in describing and exploiting the available problem information, these methods frequently produce suboptimal results. In addition, the exploration of design space is restricted. Nature-inspired optimization approaches are gaining popularity for tackling real-world issues due to its stochastic features and wider applicability to a variety of functions (without any condition of continuity or differentiability). These methods are capable of delivering high-quality solutions and resolving some of the more complicated issues that arise in real-world problems. Some examples where PSO and other have been applied to different problems occurring in industries are given in a tabular form in Table 2.1.

4 Conclusion

The basic need in different spheres of life is seeking a better solution if possible. Therefore, finding a global optimal solution for real-life problems is required for exploiting available resources to its best without wasting available human resources, money, natural resources, etc. Since most of the problems arising in various industries can be modeled as optimization problems, therefore efficient techniques are

Table 2.1 Application of PSO in manufacturing industry

Application of PSO in product processing using machining		
Production process	**Objective**	**References and methods**
Hard turning	Finding optimal value of cutting speed and feed rate, depth of cut in hard turning	[60] (NSGA-II and PSO- NN); [61] (PSO); [62] PSO)
Milling	Finding the optimal value of rotation speed, feed rate, and depth of cutting	[63] (ABC, PSO, SA); [64, 65] (PSO)
Multi-pass turning, facing, and drilling	Finding the optimal value of cutting speed, feed rate, and depth of cut	[66] (EC); [49] (PSO)
Grinding	Finding the optimal value of wheel speed, work speed, traverse speed, in feed, dress depth, and dressing lead	[67] (PSO, GSA, SCA); [68] (PSO)
High-speed machining	Finding the optimal value of bonding wear, feed per tooth, and axial depth of cut	[69] (PSO), [70] (PSO-BP neural network), [71] (PSO)
Drilling	Finding optimal value of Cutting speed, feed rate, and cutting environment	[72] (PSO); [73] (PSO)
Multi-pass turning	Finding the optimal value of cutting speed, feed rate, and depth of cut	[74] (PSO); [75] (PSO); [49] (Chaotic PSO)
Application of PSO in the paper industry		
Problem	**Objective**	**References and methods**
Paper making process	Minimizing energy cost and production rate with constrained environment. Optimizing paper making process	[76] (Advanced GA) [77] (Advanced GA); [78] (Advanced GA)
Paper making process	Minimizing trim loss and production cost	[79] (2007) (SA-PSO), [34, 80] (PSO)
Application of PSO in the production industry		
Problem	**Objective**	**References and methods**
Scheduling	Optimal scheduling of polymer batch plants; scheduling of complex products with multiple resource constraints and deep product structure; optimal power generation to short-term hydrothermal scheduling; multi-objective job-shop scheduling; trust worthy workflow scheduling in a large-scale grid with rich service resources; optimal generation schedule of the real operated cascaded hydroelectric system	[81] (PSO); [82] (PSO); [83] (Fuzzy PSO); [84] (Rotary PSO); [85] (PSO)
Production planning	Optimal production planning to meet time-varying stochastic demand; optimizing the cost of the filter, filters loss, the total demand distortion of harmonic currents, and total harmonic distortion of voltages at each bus simultaneously; assembly sequence planning of complex products; production and distribution planning of a multi-echelon unbalanced supply chain	[86] (SQP-PSO); [87] (CPSO); [88] (PSO)

needed to deal with these problems irrespective of their mathematical nature. The present study starts with a general introduction of optimization and then leads to the introduction of PSO along with its parameters, some developments, and applications in the manufacturing industry. The manufacturing industry focuses on optimizing the production processes which further benefits it in different aspects, such as increasing profits and minimizing costs/waste material. As the field is very wide, the present study covers a brief review of optimization problems arising in various industries with the aim of paving a path for implementing PSO and other nature-inspired techniques in the concerned field.

The current study is making an effort of offering research direction in the process industry using nature-inspired algorithms. The review highlights processes, objectives, process parameters, and implemented algorithms. The objectives of this chapter in brief are:

(i) To discuss the scope of particle swarm optimization algorithms for obtaining the global optimal solution of continuous as well as binary optimization problems
(ii) Developments in PSO over the decades
(iii) To provide information on various industrial processes along with objectives and parameters

References

1. Rao, S.S.: Engineering Optimization Theory and Practice, 4th edn. Wiley, Hoboken (2009)
2. Taha, H.A.: Operations Research: An Introduction, 10th ed. University of Arkansas, Fayetteville. Global Edition published by Pearson Education, England (2017)
3. Mohan, C., Deep, K.: Optimization Techniques. New Age (2009)
4. Ravindran, A., Phillips, D.T., Solberg, J.J.: Operations Research: Principles and Practice, 2nd edn. Wiley, Hoboken (2009)
5. Deb, K.: Optimization for Engineering Design: Algorithms and Examples, 2nd edn. Prentice-Hall of India Private Limited, New Delhi (1995)
6. Himmelblau, D.M.: Applied Nonlinear Programming. McGraw-Hill, New York (1972)
7. Holland, J.H.: Adaptation in Natural and Artificial System. The University of Michigan Press, Ann Arbor (1975)
8. Colorni, A., Dorigo, M., Maniezzo, V.: Distributed optimization by ant colonies. In: Proceedings of European Conference on Artificial Life (ECAL-91). Elsevier Publishing, Amsterdam (1991)
9. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: An Autocatalytic Optimizing Process, Technical Report TR91-016, Politecnico di Milano (1991)
10. Glover, F., Kochenberger, G.A.: Critical event tabu search for multidimensional knapsack problem. In: Osman, I.H., Kelly, J.P. (eds.) Meta-Heuristics: Theory and Applications, pp. 407–427. Kluwer Academic Publishers, New York (1996)
11. Passino, K.M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control. Syst. Mag. **52–67** (2002)
12. Storn, R., Price, K.: Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, Technical Report TR-95-012, Berkeley (1995)
13. Formato, R.: Central force optimization: a new nature inspired computational framework for multidimensional search and optimization. In: Nature Inspired Cooperative Strategies for

Optimization (NICSO-2007), Italy, Series: Studies in Computational Intelligence, Springer, vol. 129, pp. 221–238 (2008)
14. Formato, R.: Central force optimization: a new deterministic gradient-like optimization metaheuristic. OPSEARCH. **46**(1), 25–51 (2009)
15. Karaboga, D.: An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
16. Krishnanand, K.N., Ghose, D.: Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. Multiagent Grid Syst. **2**(3), 209–222 (2006)
17. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference Neural Networks, vol. 4, pp. 1942–1948 (1995)
18. Eberhart, R.C., Simpson, P.K., Dobbins, R.W.: Computational Intelligence PC Tools, 1st edn. Academic Press Professional, Boston (1996)
19. Clerc, M.: Think Locally, Act Locally: The Way of Life of Cheap-PSO, An Adaptive PSO, Technical Report (2001)
20. Engelbrecht, A.P.: Computational intelligence: An introduction. John Wiley and Sons, Ltd (2007)
21. Shi., Y., Eberhart, R. C.: Parameter selection in particle swarm optimization. In: Proceedings of the Seventh Annual Conference on Evolutionary Programming, New York, pp. 591–600 (1998)
22. Kennedy, J.,Eberhart, R.C.: A discrete binary version of the particle swarm algorithm. In: IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation, vol. 5, pp. 4104–4108 (1997)
23. Bergh, F., Engelbrecht, A.: A study of particle swarm optimization particle trajectories. Inform. Sci. **176**, 937–971 (2006). https://doi.org/10.1016/j.ins.2005.02.003
24. Pant, M., Thangaraj, R., Abraham, A.: Particle swarm optimization using adaptive mutation. In: Proceedings of 19th International Workshop on Database and Expert Systems Application (DEXA-2008), pp. 519–523 (2008)
25. Pant, M., Thangraj, R., Singh, V.P., Abraham, A.: Particle swarm optimization using sobol mutation. In: Proceedings of International Conference on Emerging Trends in Engineering and Technology, India, pp. 367–372 (2008)
26. Shi, Y., Eberhart, R.C.: Empirical study of particle swarm optimization. In: Proceedings of the Congress on Evolutionary Computation (CEC-1999), vol. 3, pp. 1945–1950 (1999)
27. Ratnaweera, A., Halgamuge, S., Watson, H.: Particle swarm optimization with self-adaptive acceleration coefficients. In: Proceedings of the First International Conference on Fuzzy Systems and Knowledge Discovery, pp. 264–268 (2003)
28. Suganthan, P.N.: Particle swarm optimiser with neighbourhood operator. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1958–1962 (1999)
29. Yoshida, H., Fukuyama, Y., Takayama, S., Nakanishi, Y.: A particle swarm optimization for reactive power and voltage control in electric power systems considering voltage security assessment. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, vol. 6, pp. 497–502 (1999)
30. Peram, T., Veeramachaneni, K., Mohan, C.K.: Fitness-distance-ratio based particle swarm optimization. In: Proceedings of IEEE Swarm Intelligence Symposium, pp. 174–181 (2003)
31. Venter, G., Sobieszczanski-Sobieski, J.: Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization. Struct. Multidiscip. Optim. **26**(1–2), 121–131 (2003)
32. Chauhan, P., Deep, K., Pant, M.: Novel inertia weight strategies for particle swarm optimization. Memetic Comput. **5**, 229–251 (2013)
33. Chen, G., Huang, X., Jia, J., Min, Z.: Natural exponential Inertia weight strategy in particle Swarm Optimization. In: Proceedings of 6th World Congress on Intelligent Control, pp. 3672–3675 (2006)
34. Deep, K., Chauhan, P., Pant, M.: New hybrid discrete PSO for solving non convex trim loss problem. Int. J. Appl. Evol. Comput. (IJAEC). **3**(2), 19–41 (2012)

35. Deep, K., Arya, M., Bansal, J.C.: A non-deterministic adaptive inertia weight in PSO. In: Proceedings of 13th Annual Conference on Genetic and Evolutionary Computation (GECCO-2011), ACM, New York, pp. 1155–1162 (2011)
36. Shi, Y. Eberhart, R.C., Fuzzy adaptive particle swarm optimization. In: Proceedings of IEEE Congress on Evolutionary Computation, vol. 1, pp. 101–106 (2001)
37. Liu, C., Ouyang, C., Zhu, P., Tang, W.: An adaptive fuzzy weight PSO algorithm. In: Proceedings of Fourth International conference on Genetic and Evolutionary Computing, pp. 8–10 (2010)
38. Pant, M., Thangraj, R., Singh, V.P., Particle swarm optimization using Gaussian inertia weight. In: Proceedings of International Conference on Computational Intelligence and Multimedia Applications, vol. 1, pp. 97–102 (2007)
39. Zhu, H., Zheng, C., Hu, X., Li, X.: Adaptive PSO using random inertia weight and its application in UAV path planning. In: Proceedings of Seventh International Symposium on Instrumentation and Control Technology: Measurement Theory and Systems and Aeronautical Equipment (SPIE), 7128, pp. 1–5 (2008)
40. Chen, J.Y., Shen, J.J.: Structure learning of Bayesian network using a chaos-based PSO. Adv. Mater. Res. **2292–2295** (2012)
41. Feng, Y., Yao, Y.M., Wang, A.: Comparing with chaotic inertia weights in particle swarm optimization. In: Proceedings of International Conference on Machine Learning and Cybernetics, pp. 329–333 (2007)
42. Bansal, J.C., Singh, P.K., Saraswat, M., Verma, A., Jadon, S.S., Abraham, A.: Inertia weight strategies in particle swarm optimization. In: Proceedings of Third World Congress on Nature and Biologically Inspired Computing (NaBIC-2011), pp. 633–640 (2011)
43. Nickabadi, A., Ebadzadeh, M.M., Safabakhsh, R.: A novel particle swarm optimization algorithm with adaptive inertia weight. Appl. Soft Comput. **11**(4), 3658–3670 (2011)
44. Deep, K., Chauhan, P., Pant, M.: Totally disturbed chaotic Particle Swarm Optimization. In: 2012 IEEE Congress on Evolutionary Computation, pp. 1–8 (2012)
45. Xie, X., Zhang, W., Yang, Z.: A dissipative particle swarm optimization. In: Proceedings of IEEE Congress on Evolutionary Computation (CEC-2002), pp. 1456–1461 (2002)
46. Alatas, B., Akin, E., Ozer, A.B.: Chaos embedded particle swarm optimization algorithms. Chaos, Solitons Fractals. **40**(4), 1715–1734 (2009)
47. Liu, B., Wang, L., Jin, Y.H., Tang, F., Huang, D.X.: Improved particle swarm optimization combined with chaos. Chaos, Solitons Fractals. **25**(5), 1261–1271 (2005)
48. He, Q., Han, C.: An improved particle swarm optimization algorithm with disturbance term. ICIC. **3**, 100–108 (2006)
49. Chauhan, P., Pant, M., Deep, K.: Parameter optimization of multi-pass turning using chaotic PSO. Int. J. Mach. Learn. Cybern. **6**, 319–337 (2015)
50. Li, C., Zhou, J., Kou, P., Xiao, J.: A novel chaotic particle swarm optimization based fuzzy clustering algorithm. Neurocomputing **83**, 98–109 (2012)
51. Mukhopadhyay, S., Banerjee, S.: Global optimization of an optical chaotic system by chaotic multi swarm particle swarm optimization. Expert Syst. Appl. **39**(1), 917–924 (2012)
52. Deep, K., Bansal, J.C.: A modified binary particle swarm optimization for knapsack problems. Appl. Math. Comput. **218**(22), 11042–11061 (2012)
53. Marandi, A., Afshinmanesh, F., Shahabadi, M., Bahrami, F.: Boolean particle swarm optimization and its application to the design of a dual-band dual-polarized planar antenna. In: Proceedings of IEEE Congress on Evolutionary Computation (CEC-2006), pp. 3212–3218 (2006)
54. Singh, Y., Chauhan, P.: New mutation embedded generalized binary PSO. In: Sathiyamoorthy, S., Caroline, B., Jayanthi, J. (eds.) Emerging Trends in Science, 2012, Engineering and Technology Lecture Notes in Mechanical Engineering, pp. 705–715. Springer, New Delhi (2012)

55. Lee, S., Park, H., Jeon, M.: Binary Particle swarm optimization with bit change mutation. In: IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E90-A:10, pp. 2253–2256 (2007)
56. Chauhan, P., Pant, M., Deep, K.: Novel binary PSO for continuous global optimization problems. In: Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011) December 20–22, p. 130 (2011)
57. Deep, K., Chauhan, P., Pant, M.: Multi task selection including part mix, tool allocation and process plans in CNC machining centers using new binary PSO. In: 2012 IEEE Congress on Evolutionary Computation, pp. 1–8 (2012)
58. Chauhan, P., Pant, M., Deep, K.: Gompertz PSO variants for Knapsack and Multi-Knapsack problems. Appl. Math. J. Chin. Univ. **36**, 611–630 (2021)
59. Unler, A., Murat, A.: A discrete particle swarm optimization method for feature selection in binary classification problems'. Eur. J. Oper. Res. **206**(3), 528–539 (2010)
60. Bouacha, K., Terrab, A.: Hard turning behavior improvement using NSGA-II and PSO-NN hybrid model. Int. J. Adv. Manuf. Technol. **86**, 3527–3546 (2016)
61. Omkar, M., Chinchanikar, S., Gadge, M.: Multi-performance optimization in hard turning of AISI 4340 steel using particle swarm optimization technique. Mater. Today Proc. **5**(11 Part 3), 24652–24663 (2018)
62. Mishra, R.R., Kumar, R., Panda, A., Pandey, A., Sahoo, A.K.: Particle swarm optimization of multi-responses in hard turning of D2 steel. In: Das, H., Pattnaik, P., Rautaray, S., Li, K.C. (eds.) Progress in Computing, Analytics and Networking. Advances in Intelligent Systems and Computing, vol. 1119. Springer, Singapore (2020)
63. Rao, R.V., Pawar, P.J.: Parameter optimization of a multi-pass milling process using non-traditional optimization algorithms'. Appl. Soft Comput. **10**(2), 445–456 (2010)
64. Bahirje, S., Potdar, V.: Review paper on implementation of particle swarm optimization for multi-pass milling operation. Int. J. Eng. Res. Technol. (IJERT). **9**(9), 237–239 (2020)
65. Farahnakian, M., Razfar, M.R., Moghri, M., Asadnia, M.: The selection of milling parameters by the PSO-based neural network modeling method. Int. J. Adv. Manuf. Technol. **1–12** (2011)
66. Sankar, R.S., Asokan, P., Saravanan, R., Kumanan, S., Prabhaharan, G.: Selection of machining parameters for constrained machining problem using evolutionary computation. Int. J. Adv. Manuf. Technol. **32**(9–10), 892–901 (2007)
67. Shin, T., Adam, A., Abidin, A.: A comparative study of PSO, GSA and SCA in parameters optimization of surface grinding process. Bull. Electr. Eng. Inf. **8**(3), 1117–1127 (2019)
68. Pawar, P.J., Rao, R.V., Davim, J.P.: Multiobjective optimization of grinding process parameters using particle swarm optimization algorithm. Mater. Manuf. Process. **25**(6), 424–431 (2010)
69. Abbas, A.T., Sharma, N., Anwar, S., Hashmi, F.H., Jamil, M., Hegab, H.: Towards optimization of surface roughness and productivity aspects during high-speed machining of Ti–6Al–4V. Materials. **12**(22), 3749 (2019)
70. Zheng, J.X., Zhang, M.J., Meng, Q.X.: Tool cutting force modeling in high speed milling using PSO-BP neural network. In: Key Engineering Materials, vol. 375, pp. 515–519. Trans Tech Publications, Ltd. (2008)
71. Cus, F., Zuperl, U., Gecevska, V.: High speed end-milling optimisation using Particle Swarm Intelligence. J. Achieve. Mater. Manuf. Eng. **22**(2), 75–78 (2007)
72. Kumar, S.M.G., Jayaraj, D., Kishan, A.R.: PSO based tuning of a PID controller for a high performance drilling machine. Int. J. Comput. Appl. **1**(19), 12–18 (2010)
73. Gaitonde, V.N., Karnik, S.R.: Minimizing burr size in drilling using artificial neural network (ANN)-particle swarm optimization (PSO) approach. J. Intell. Manuf. **23**, 1783–1793 (2012)
74. Bharathi, R.S., Baskar, N.: Particle swarm optimization technique for determining optimal machining parameters of different work piece materials in turning operation. Int. J. Adv. Manuf. Technol. **54**(5–8), 445–463 (2011)
75. Yusup, N., Zain, A.M., Hashim, S.Z.M.: Overview of PSO for optimizing process parameters of machining. Proc. Eng. **29**, 914–923 (2012)

76. Santos, A., Dourado, A.: Global optimization of energy and production in process industries: a genetic algorithm application. Control. Eng. Pract. **7**, 549–554 (1999)
77. Wang, H., Borairi, M., Roberts, J.C., Xiao, H.: Modelling of a paper making process via genetic neural networks and first principle approaches. In: Proceedings of the IEEE International Conference on Intelligent Processing Systems, ICIPS, Beijing, pp. 584–588 (1997)
78. Borairi, M., Wang, H., Roberts, J.C.: Dynamic modelling of a paper making process based on bilinear system modelling and genetic neural networks. In: Proceedings of UKACC International Conference on Control, pp. 1277–1282 (1998)
79. Xianjun, S., Li, Y., Zheng, B., Dai, Z.: General particle swarm optimization based on simulated annealing for multi-specification one-dimensional cutting stock problem. In: Computational Intelligence and Security. Springer-Verlag Berlin, Heidelberg, LNAI, pp. 67–76 (2007)
80. Deep, K., Chauhan, P., Bansal, J.C.: Solving nonconvex trim loss problem using an efficient hybrid Particle Swarm Optimization. In: World Congress on Nature & Biologically Inspired Computing (NaBIC), pp. 1608–1611 (2009)
81. Hota, P.K., Barisal, A.K., Chakrabarti, R.: An improved PSO technique for short-term optimal hydrothermal scheduling. Electr. Power Syst. Res. **79**(7), 1047–1053 (2009)
82. Sha, D.Y., Lin, H.-H.: A multi-objective PSO for job-shop scheduling problems. Expert Syst. Appl. **37**(2), 1065–1070 (2010)
83. Liu, H., Abraham, A., Hassanien, A.E.: Scheduling jobs on computational grids using fuzzy particle swarm algorithm. Futur. Gener. Comput. Syst. **26**, 1336–1343 (2010)
84. Tao, Q., Chang, H., Yi, Y., Gu, C., Li, W.: A rotary chaotic PSO algorithm for trustworthy scheduling of a grid workflow. Comput. Oper. Res. **38**(5), 824–836 (2008)
85. Mahor, A., Rangnekar, S.: Short term generation scheduling of cascaded hydro electric system using novel self adaptive inertia weight PSO. Int. J. Electr. Power Energy Syst. **34**(1), 1–9 (2012)
86. Chang, Y.P.: Integration of SQP and PSO for optimal planning of harmonic filters. Expert Syst. Appl. **37**(3), 2522–2530 (2010)
87. Wang, Y., Liu, J.H.: Chaotic particle swarm optimization for assembly sequence planning. Robot. Comput. Integr. Manuf. **26**(2), 212–222 (2010)
88. Che, Z.H.: A particle swarm optimization algorithm for solving unbalanced supply chain planning problems. Appl. Soft Comput. **12**(4), 1279–1287 (2012)