International Journal of
Computer Aided Manufacturing

ISSN: 2456-642X
Volume 11, Issue 1, 2025
January–June
DOI (Journal): 10.37628/IJCAM

https://journalspub.com/journal/IJCAM/

Review IJCAM

Particle Swarm Optimization Algorithm and Its Developments: A Review

Sourav Rajak[1,*], Sarnendu Paul[2], Srijan Paul[2], Suraj Yadav[2], Kaushal Kishore[2]

Abstract

Fish schools and bird flocks are examples of natural phenomena that served as inspiration for the stochastic optimization method known as particle swarm optimization (PSO). According to the report, PSO has evolved since its introduction in 1995 because of researchers' constant efforts to enhance and adapt the algorithm to meet different requirements and uses. Beginning with its origins and history, the paper covers several PSO-related subjects. With an emphasis on several topics, such as algorithm structure, parameter selection, topological structure, discrete PSO, parallel PSO, multi-objective optimization PSO, and engineering applications, the article examines the present status of PSO research and applications. This thorough examination demonstrates PSO's adaptability and broad range of applications in several fields. All things considered, this work seems to be a useful tool for comprehending the theory, history and current state of PSO.

Keywords: Particle swarm optimization, discrete swarm optimization, multi-objective optimization, PSO, Inertia Weight

INTRODUCTION TO PSO

Optimization problems are ubiquitous across scientific, engineering, and industrial domains, arising in tasks, such as resource allocation, process optimization, scheduling, and machine learning. These problems are often characterized by complex, multi-dimensional search spaces that make traditional optimization techniques inadequate. Particle Swarm Optimization (PSO), introduced by Kennedy and Eberhart in 1995, revolutionized the field by offering a robust and versatile approach to tackle such challenges.

Unlike deterministic methods, PSO leverages stochastic processes and swarm intelligence, which allow it to navigate complex, non-linear, and multi-modal landscapes with efficiency. Its simplicity, ease of implementation, and adaptability have contributed to its widespread adoption in solving real-world problems. This paper aims to provide a detailed exploration of PSO, focusing on:

- The theoretical foundations that underpin its operation and principles.
- Its diverse range of applications across scientific, industrial, and engineering domains, demonstrates its versatility and effectiveness.
- Key algorithmic variations developed over the years to enhance performance and address specific challenges [1].
- Existing limitations, including issues, like premature convergence and sensitivity to parameter settings, affect its optimization capabilities.
- Future research directions that can pave the way for advancing PSO and broadening its scope in emerging areas, such as artificial intelligence, quantum computing, and sustainability-oriented optimization problems.

*__Author for Correspondence__
Sourav Rajak
E-mail: sarnendupaul1989@gmail.com

[1]Student, Department of Mechanical Engineering, Asansol Engineering College, Asansol, West Bengal, India
[2]Faculty, Department of Mechanical Engineering, Asansol Engineering College, Asansol, West Bengal, India

Received Date: February 14, 2025
Accepted Date: February 28, 2025
Published Date: March 11, 2025

Citation: Sourav Rajak, Sarnendu Paul, Srijan Paul, Suraj Yadav, Kaushal Kishore. Particle Swarm Optimization Algorithm and Its Developments: A Review. International Journal of Computer Aided Manufacturing. 2025; 11(1): 1–5p.

THEORETICAL FOUNDATION OF PSO

PSO is inspired by the collective behavior observed in nature, such as flocking birds and schooling fish. This natural phenomenon showcases how individual agents, with limited intelligence, work collectively to achieve a common goal. In the case of PSO, particles in the swarm mimic such behaviors to explore and exploit the search space effectively. These particles, representing potential solutions, communicate and collaborate to find optimal solutions by learning from their individual experiences and the overall performance of the group [2–4].

The algorithm operates on the following principles:

- *Swarm Intelligence:* The collective behavior of a decentralized, self-organized system where each particle represents a potential solution and contributes to the swarm's success.
- *Particle Movement:* Each particle adjusts its position iteratively, balancing its own historical knowledge (local best) with the guidance from the best-performing particle in the swarm (global best). This movement ensures diversity in the search process and prevents stagnation at suboptimal solutions.
- *Fitness Function:* A predefined objective function evaluates the quality of each particle's position, guiding the swarm toward areas of higher fitness within the search space. The fitness value serves as a measure of how "good" a solution is.

The velocity and position of each particle are updated using:

- where; Inertia weight controlling the trade-off between exploration (global research) and exploitation (local refinement).
- Cognitive and social acceleration coefficients, which determine the relative influence of the particle's own experience and the swarm's collective wisdom.
- Random numbers between 0 and 1, introducing stochasticity to enhance the algorithm's robustness.
- The particle's best-known position so far, represents the optimal solution found by the particle itself.
- The global best-known position of the swarm represents the best solution discovered by any particle in the swarm.

This combination of local and global information sharing allows PSO to efficiently navigate complex, high-dimensional search spaces. By continuously refining their positions and velocities, particles converge toward optimal or near-optimal solutions. However, striking the right balance between exploration and exploitation remains a critical aspect of PSO's design [5].

FUNCTION OF PSO

Since the PSO's introduction, other variants have been developed. Before introducing two alternatives, we will only go into detail on how the original one operates. Remember that the main idea is that we have swarming particles moving around a problem space while a fitness function is used to determine their placements.

According to this PSO approach, a set of particles is seeded into the issue space after it has been described, and each particle's location and velocity are continuously updated. Although PSO has been shown to be a successful algorithm with favorable results, it is not by designing a method that guarantees the discovery of the best solution because it depends on visits and evaluating the locations of issue spaces. Although there are many kinds, the majority all serve a fitness function. Because its definition depends on the problem being optimized (especially in terms of its dimensions), we will just call this function f(xi), which is short for f(xi,0,..., f(xi,d). This function displays how well the "I" particle is positioned in the multidimensional space with respect to the desired result. These weights and binds the D dimensions to be optimized given a problem that is described as an optimization in one of the D dimensions. Because the technique is multi-dimensional, the positions and velocities of the particles we controlled will have d components; hence, we have positions as xi(xi,0,..., xi,d) and velocities as vi(vi,0,..., vi,d).

GLOBAL BEST

In our approach, we have a completely connected swarm, where all the particles exchange information and each particle know the optimal location that every other particle in the swarm has ever been to.

Each particle's position (1) and velocity (2) are calculated as follows:

Mathematical Framework

Position Update: $x(t + 1) = x(t) + v(t + 1)$ 1

Velocity Update: $v(t + 1) = w * v(t) + c1 * r1 * (pbest - x(t)) + c2 * r2 * (gbest – x(t))$ 2

where:

- x(t): Current position.
- v(t): Current velocity.
- w: Inertia weight (typically 0.4–0.9).
- c1, c2: Acceleration coefficients.
- r1, r2: Random numbers (0–1).

To find their placements, all the swarm's particles are dispersed as needed over the search space, which can be done either uniformly or at random. Although a random integer is commonly used to set the velocity, lower values are usually preferred to avoid large initial offsets. The values of the social and cognitive component scales are also selected at initialization time because they do not change during the optimization process.

Algorithm 2 displays the pseudo-code for the basic PSO approach. Although there are many other options available, each of which has a different impact on the algorithm's performance, we provide a simple solution to several problems that were not fully addressed in the initial PSO presentation [3]. These elements consist of the genuine PSO algorithm's halting condition and method 1's particle positions and velocities.

When the Local version of the algorithm was introduced by the same author, he mentioned it for both versions to prevent particles from flying out of the search space [1]. The main parameterization needed with this algorithm relates to the acceleration multipliers and the maximum velocity (velocity clamping). You may have noticed that in addition to the algorithm's parameters, you must also specify the search space's bounds and the algorithm's maximum number of iterations. However, these parameters should be directly derived from the problem being modelled, and the number of iterations can be easily set for any reasonable value reported in the literature, such as in the modified PSO [6].

LOCAL BEST

This innovation decreases the neighborhood where particles share information compared to the original method. The swarm is separated into neighborhoods where particles only share their best value with their neighbors, rather than each particle knowing the best value globally. However, neighborhoods must overlap for convergence to be most effective on a global scale. Because of this difference, this version of the method covers a larger portion of the search space and is less likely to have local minimum, but it takes longer to find a solution [1].

Although slower convergence speed is the price paid for improved search space coverage, it increases the likelihood that the best solution will be found in the PSO context. Consequently, it can require more iteration than the initial one to truly come to a consensus.

There are two easy approaches to build the neighborhoods: dynamically by geographic distance (which is more expensive to process) or statically by particle index. Nonetheless, there are other social network topologies that specify particle distributions in neighborhoods, which are known to affect the algorithm's performance [3].

INERTIA WEIGHT

This method aims to balance these two PSO inclinations (depending on parameterization) by exploring new parts of the search space or by utilizing areas close to current solutions. To do this, this variant focusses on the momentum component of equation 2 for the particle's velocity. You will see that when this component is eliminated, the particle's motion loses sight of its previous trajectory and always explores close to a solution. If the velocity component is used or even multiplied by an inertial weight (w) factor, the momentum component's relevance is balanced. The particle will unavoidably attempt new regions of the search space due to its limits, rapidly changing its speed in the direction of the particle will unavoidably explore new areas of the search space because of its limits, and it will swiftly change its speed in the direction of the best answers [7–15]. It needs to "counteract" initially. In places where the "time spent" "counteracts" the push from earlier, it permits the study of new concepts while preserving the pace already achieved. This variation is generated worth by applying a weight to the previous velocity component. The studies conducted by the author have demonstrated that in. In the range [0.9, 1.2], the w value should be as high as feasible to strike a good compromise between performance benefit and the algorithm's ability to locate the desired resolution. Further investigation showed that a variable w value significantly increased this ratio; the studied example had a linear w drop [6].

Because fixed inertia weights often do not yield good results, numerous PSO versions have been devised. A quadratic function, population information, Bayesian techniques, adaptive changes, and exponential decreasing inertia weight strategy are some of these variations [16–18].

FUTURE AIMS

To find any limitations and apply the technique properly, future studies will search for the algorithm's shortcomings. Basic uni-modal functions are one example of this problem, where regrouping is not necessary because particles rapidly and precisely approximate the global minimizer and no superior minimizer can be found. There does not seem to be anything that would prevent the regrouping mechanism from being utilized with a different search algorithm as its foundation, even if it has been tested using the best conventional PSO to demonstrate its usefulness. When combined with a superior local minimizer, such as GCPSO, performance may be improved. Due to the lack of systematic study on heterogeneity in PSO, there are still gaps in our understanding of how heterogeneity affects PSO algorithms. The development of novel particle swarm optimization methods have a lot of promises.

Appropriate parameter and topology selection is also one of the PSO study areas.

CONCLUSIONS

In conclusion, social psychologist James Kennedy and electrical engineer Russell C. Eberhart created Particle Swarm Optimization (PSO). Based on the idea of swarm intelligence, this method optimizes solutions by imitating the actions of swarming animals. The basic idea behind PSO is that particles move around a problem space, assess their positions using a fitness function, and modify their velocities in response to the swarm's and their own experiences. While the best local version of the algorithm restricts the sharing to particle neighborhoods, the best global version guarantees that all particles exchange information about the best solutions discovered.

Exploration and exploitation are balanced in PSO by the weight of inertia. The technique's adaptability, variants, and parameter configurations enhance its efficacy in resolving optimization issues, and current studies seek its constraints and provide fresh methods.

REFERENCES

1. Eberhart RC, Kennedy J. A new optimizer using particle swarm theory. In Proceedings of the sixth international symposium on micro machine and human science. New York, NY, USA: IEEE. 1995;43:6.
2. Eberhart RC, Shi Y. Particle swarm optimization: Developments, applications and resources. In Proceedings of the 2001 congress on evolutionary computation. Piscataway, NJ, USA: IEEE; 2001;1:81–86.
3. A.P. Engelbrecht. Computational intelligence: An introduction. Wiley; 2007.
4. Kennedy J, Eberhart RC. Particle swarm optimization. In Proceedings of IEEE international conference on neural networks. Perth, Australia. 1995;4:1942–1948.
5. Poli R. An analysis of publications on particle swarm optimization applications. Essex, UK: Department of Computer Science, University of Essex; 2007.
6. Shi Y, Eberhart R. A modified particle swarm optimizer. In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference. IEEE; 2002. p. 69–73.
7. Abdelbar AM, Abdelshahid S, Wunsch DCI. Fuzzy pso: A generalization of particle swarm optimization. In: Proceedings of 2005 IEEE international joint conference on neural networks (IJCNN ’05) Montreal, Canada; July 31–August 4, 2005. p. 1086–1091.
8. Acan A, Gunay A. Enhanced particle swarm optimization through external memory support. In: Proceedings of 2005 IEEE congress on evolutionary computation, Edinburgh, UK.; Sept 2–4, 2005. p. 1875–1882.
9. Afshinmanesh F, Marandi A, Rahimi-Kian A. A novel binary particle swarm optimization method using artificial immune system. In: Proceedings of the international conference on computer as a tool (EUROCON 2005) Belgrade, Serbia; Nov 21–24, 2005. p. 217–220.
10. Al-kazemi B, Mohan CK. Multi-phase generalization of the particle swarm optimization algorithm. In: Proceedings of 2002 IEEE Congress on Evolutionary Computation, Honolulu, Hawaii. August 7–9, 2002. p. 489– 494.
11. Rifaie MM, Blackwell T. Bare bones particle swarms with jumps ants. Lect Notes Comput Sci Ser. 2012;7461(1):49–60.
12. Ardizzon G, Cavazzini G, Pavesi G. Adaptive acceleration coefficients for a new search diversification strategy in particle swarm optimization algorithms. Inf Sci. 2015;299:337–378.
13. Banka H, Dara S. A hamming distance based binary particle swarm optimization (HDBPSO) algorithm for high dimensional feature selection, classification and validation. 2015.
14. Ivatloo BM. Combined heat and power economic dispatch problem solution using particle swarm optimization with time varying acceleration coefficients. Electr Power Syst Res. 2013;95(1):9–18.
15. Jamian JJ, Mustafa MW, Mokhlis H. Optimal multiple distributed generation output through rank evolutionary particle swarm optimization. Neurocomputing. 2015;152:190–198.
16. Jia D, Zheng G, Qu B, Khan MK. A hybrid particle swarm optimization algorithm for high-dimensional problems. Comput Ind Eng. 2011;61:1117–1122.
17. Jian W, Xue Y, Qian J. An improved particle swarm optimization algorithm with neighborhoods topologies. In: Proceedings of 2004 international conference on machine learning and cybernetics, Shanghai, China; August 26–29, 2004. p. 2332–2337.
18. Jiang CW, Bompard E. A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization. Math Comput Simul. 2005;68:57–65.