International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN (P): 2249-6831; ISSN (E): 2249-7943
Vol. 8, Issue 2, Jun 2018, 37-48

TRANS STELLAR
•Journal Publications •Research Consultancy

Original Article

PARTICLES SWARM OPTIMIZATION TECHNIQUES : PRINCIPLE, COMPARISON & APPLICATION

P. SANYASI NAIDU[1], BABITA BHAGAT[2] & NEHA RATHI[3]

[1]*Department of Computer Science & Engineering, GITAM University, Visakhapatnam, Andhra Pradesh, India*

[2]*Department of Computer Engineering, AP PHCET, Rasayani, Maharashtra, India*

[3]*Department of Computer Engineering, MH PHCET, Rasayani, Maharashtra, India*

ABSTRACT

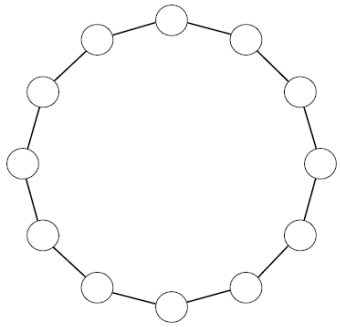
Particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It solves a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best -known position (pbest), but is also guided toward the best -known positions (gbest) in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions. The particles move in the search space with considering its own velocity and position called as pbest, but pbest has the tendency to flow around the local optima. Because of this problem we compare the different Particle swarm optimization based algorithm with its principles & application in this paper. Variable Neighbourhood PSO, Adaptive PSO & Niche PSO compare to see the performance of the particles in the search space with respect to time.

KEYWORDS: Cloud Computing, Metaheuristic, VNPSO, NPSO & APSO

Received: Apr 05, 2018; **Accepted:** Apr 26, 2018; **Published:** May 08, 2018**; Paper Id.:** IJCSEITRJUN20185

1. INTRODUCTION

Particle swarm optimization is a stochastic population-based optimization approach, first published by Kennedy and Eberhart in 1995. A large body of research has been done to study the performance of PSO, and to improve its performance. Much effort has been invested to obtain a better understanding of PSO.

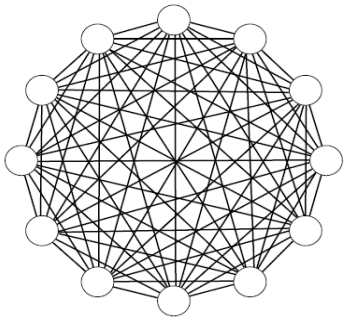


Graphical representation of the *lbest* model

Figure 1

These studies concentrated mostly on a better understanding of the basic PSO control parameters, namely the acceleration coefficients, inertia weight, velocity clamping, and swarm size. From these empirical studies, it can

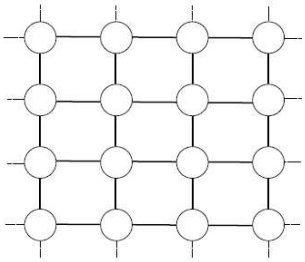
be concluded that the PSO is sensitive to control parameter choices, specifically the inertia weight, acceleration coefficients and velocity clamping. Wrong initialization of these parameters may lead to divergent or cyclic behavior.



Graphical representation of the *gbest* model

Figure 2

To gain a better, general understanding of the behavior of particle swarms, in-depth theoretical analyses of particle movement in search space is necessary. A few theoretical studies of particle movement can be found, which concentrate on simplified PSO systems.



Graphical representation of the von Neumann model

Figure 3

2. LITERATURE SURVEY

In, James Kennedy and Russell Eberhart **[17]** Particle swarm optimization as developed by the authors comprises a very simple concept, and paradigms can be implemented in a few lines of computer code. It requires only primitive mathematical operators and is computationally inexpensive in terms of both memory requirements and speed. Early testing has found the implementation to be effective with several kinds of problems. This paper discusses the application of the algorithm to the training of artificial neural network weights.

In, Hui Zhan, Jun Zhang [15] An adaptive particle optimization (APSO) that features better search efficiency that classical particle swarm optimization (PSO) is presented. More importantly, it can perform a global search over the entire search space with faster convergence speed. The APSO consists of two steps, including exploration, exploitation, convergence and jumping out in each generation. It enables the automatic control of inertia weight, acceleration coefficient, and other algorithm parameters at runtime to improve the search efficiency and convergence speed.

In, Julio Barrera and Carlos A. Coello-Coello [28], we will review the most representative PSO-based approaches that have been proposed to deal with multimodal optimization problems. Such approaches include the simple introduction of powerful mutation operators, schemes to maintain diversity that was originally introduced in the genetic algorithms literature (e.g., niching), the exploitation of local topologies, the use of species, and clustering, among others.

In Pierre Hansen, Nenad Mladenovic [27]they examine a relatively unexplored approach to the design of heuristics, change of neighborhood in the search. Using semantically this idea and very little more i.e only a local search routine leads to a new metaheuristic, which is widely applicable. We call this approach variable neighborhood search (VNS). Contrary to other metaheuristic based on local search method, VNS does not follow a trajectory but explores increasingly distant neighborhoods of the current incumbent solution, and jumps from this solution to a new one if and only if an improvement has been made.

3. PSO

Canonical PSO Model

The canonical PSO algorithm was inspired from the behavior of bird flocking and fish schooling [1] [2] [3]. The two main factors that characterize the status of a particle in the search space are its position and velocity. Eq. (1) is used to update the velocity and Eq. (2) is used to determine the new position by adding the previous position and the new velocity [7].

$$h_{iq}(t+1) = Ih_{iq}(t) + ac_1 r_1\left(g_{iq}^{\neq}(t) - g_{iq}(t)\right) + ac_2 r_2\left(g_q^{*}(t)\right) - g_{iq}(t)\ ) \tag{1}$$

$$g_{iq}(t+1) = g_{iq}(t) + h_{iq}(t+1) \tag{2}$$

In eq. (1) and eq. (2), h_{iq}, g_{iq}, $g_{iq}^{\neq}(t)$ and $g_q^{*}(t)$ are the velocity of particle i, a position of the particle, previous best position and the best location among all particles in the population respectively. Here r_1 and r_2 are the random functions within a range $[0,1]$ and ac_1 and ac_2 are the two positive constant parameters called acceleration coefficients, which reduce the maximum step size of the particle. The desirable results can be obtained by selecting $ac_1 \geq ac_2$ with $ac_1 + ac_2 \leq 4$ and I denotes the inertia factor, while progress the algorithm its value gets reduces from 1.0 to 0.

In order to monitor the particle in search space area, the maximum movement of velocity in one iteration and the position of the particle is limited within the range as in eq. (3) and eq. (4) as,

$$h_{iq} = sign(h_{iq})\min\left(|h_{iq}|, h_{\max}\right) \tag{3}$$

$$g_{i,q} = sign(g_{i,q})\min\left(|g_{i,q}|, g_{\max}\right) \tag{4}$$

Here, $h_{\max}$ is the maximum velocity and $g_{\max}$ is the maximum solution variable.

Algorithm 1: The Pseudo Code for PSO Algorithm is explained in Algorithm 1 as Follows:

Algorithm 1: Pseudo Code of PSO Algorithm

Input: N, I, ac_1 and ac_2, r_1 and r_2, h_i, g_i.
Output: *gbest*
Begin
Initialize N number of particles. For each particle $i \in N$,
Initialize velocity h_i and position g_i randomly Evaluate g_i by determining $f(g_i)$ using eq. (6) if $f(g_i) < f(pbest_i)$ for each particle i, then

Algorithm 1: Contd.,
$pbest_i = g_i$ for each particle i
end if
if $\min\{f(pbest_i)\} \forall\ i < f(gbest)$ then
$gbest = pbest_i^*$
end if
Update the particle's position and velocity using eq. (7) and eq. (8)
end for
$t = t+1$
Until $t > Max\_iterations$

Generally, the PSO algorithm [4] [5] [6] has two basic updating models namely gbest model and pbest model. According to eq. (1), the global best (gbest) position $g_q^*(t)$ is the best particle (obtained by means of fitness function) till the t^{th} iteration and the local best solution (pbest) $g_{iq}^{\neq}(t)$ is the best i^{th} particle in the t^{th} iteration and it is given in eq. (11),

$$g_i^{\neq} = \arg\min_{i=1}^{N}\left(f\left(g_i^{\neq}(t-1)\right), f\left(g_i(t)\right)\right) \tag{5}$$

The gbest and pbest along with r_1 and r_2 control the effect of new particle velocity. Generally, gbest model converges quickly than the pbest model, and also it has the tendency to confine in local optima but pbest model has the tendency to flow around the local optima.

Variable Neighbourhood Particle Swarm Optimization Algorithm (VNPSO)

VNPSO algorithm, which is a recent metaheuristic algorithm [7] inspired from VNS that depends on the growing size of the neighborhood to find the better local optima using some strategies. VNS getaway from the problem of current local minimum g^* by beginning the local searches from the initiating points experienced from the neighborhood of g^*, that increases its size repeatedly until it founds better local minima than the current one. If the velocity of the particle is dropped below the threshold ve_t, then the new velocity is set using the eq. (6),

$$ve_{iq}(t) = Wve_1 + ac_1r_1\left(g_{iq}^{\neq}(t-1) - g_{iq}(t-1)\right) + ac_2r_2\left(g_q^*(t-1) - g_{iq}(t-1)\right) \tag{6}$$

$$ve_1 = \begin{Bmatrix} ve_{iq} & if & |ve_{iq}| \geq ve_t \\ u_d ve_{\max} \big/ \rho & if & |ve_{iq}| < ve_t \end{Bmatrix} \tag{7}$$

In eq. (7), u_d represents uniform random distribution function, $ve_{\max}$ is the maximum velocity and ve_{iq} is the velocity of i^{th} particle with a dimension q. The value of two parameters ve_t and ρ are correlated with the performance of the algorithm. If ve_t is large, then the period of oscillation gets reduced and it increases the probability that the particles jump over the local minima for the same number of iterations and also it prevents the particle from converging by keeping the particles in the fast flying state. These processes are repeated until it satisfies the termination criteria.

The pseudo code of VNPSO is summarized in given as Algorithm 2 in the paper.

Algorithm 2: Pseudo code Of VNPSO Algorithm

Initialize N number of particles.
For each particle $i \in N$ and each dimension q ,
Initialize velocity h_i and position g_i randomly. Set $Nohope = 0$ // Flag of iterations without improvement
If the end criteria is not met,
Evaluate each particle's position depending upon their fitness function
If the global best position $g^*(t)$ is improved, then $Nohope = 0$, or else, $Nohope = Nohope + 1$
Set the local best position,
end if
If $Nohope < 10$, then
Update the particle's position and velocity
Else
Update the particle's position and velocity
end if
end if $t = t + 1$ Until $t > Max\_iteration$

Adaptive Particle Swarm Optimization Algorithm (APSO)

Accelerating convergence speed and avoiding the local optima have become the two most important and appealing goals in PSO research. However, so far, it is seen to be difficult to simultaneously achieve both goals. For example, the comprehensive-learning PSO (CLPSO) in [8] focuses on avoiding the local optima but brings in a slower convergence as a result. To achieve both goals, adaptive PSO (APSO) is formulated. Adaptive parameter control strategies can be developed based on the identified evolutionary state and by making use of existing research results on inertia weight [9]–[12] and acceleration coefficients [13]–[16]. It can perform a global search over the entire search space with faster convergence speed. The APSO consists of two main steps. First, by evaluating the population distribution and particle fitness, areal-time evolutionary state estimation procedure is performed to identify one of the following four defined evolutionary states, including exploration, exploitation, convergence, and jumping out in each generation. It enables the automatic control of inertia weight, acceleration coefficients, and other algorithmic parameters at runtime to improve the search efficiency and convergence speed. Then, an elitist learning strategy is performed when the evolutionary state is classified as convergence state. The strategy will act on the globally best particle to jump out of the likely local optima.

$$v_i^d = wv_i^d + c_1 rand_1^d(pBest_i^d - x_i^d) + c_2 rand_2^d(nBest_i^d - x_i^d) \quad (8)$$

$$x_i^d = x_i^d + v_i^d \quad (9)$$

During the evolutionary process, the velocity and position of particle ion dimension d are updated as where ω is the inertia weight [9], c1 and c2 are the acceleration coefficients [2], and randd1 and randd2 are two uniformly distributed random numbers independently generated within [0, 1] for the d^{th} dimension [1].

Algorithm 3: Pseudo Code of APSO Algorithm

```
Procedure Proposed Algorithm
begin
Initialization swarms
Repeat
adopt swarms number according to free swarms
for each swarm do
 if a change is detected in the environment then
evaluate all pbests then regenerate particles randomly in a
hyper sphere with radius r centered at gbest and update
pbests
update gbest
 else
 if swarm is the best swarm
do local search around gbest in hyper sphere with radius r
and update only gbest
end if
For each particle in swarm do
update particle position
update pbest and gbest
end-for
end-if
test for converge
end-for
r_excl=fcmdist
for each pair of swarms m and n, m<>n do
test for conflict if there is delete worse swarm
end-for
until a maximum number of fitness evaluations is reached
 end.
```

Niche Particle Swarm Optimization Algorithm (NPSO)

The NichePSO algorithm was proposed in [12]. This approach uses the Guaranteed Convergence Particle Swarm Optimizer from van den Bergh and Engelbrecht [13] together with the gbest model, aiming to prevent premature convergence of the particles. Additionally, the cognition only model is used to encourage the local search of the particles. The method used to identify niches in the NichePSO algorithm is based on keeping track of the changes in the fitness value of the particles. If a particle does not show any change within a certain number of iterations, a niche is created, containing the particle and its closest topological neighbor. The Euclidean distance is used to determine the closeness of the particles so that the closest topological neighbor to the particle selected is the particle within the minimum distance to it. A niche radius is computed for each sub-swarm. This radius is the maximum distance between the particles in the sub-swarm and is described in equation (10).

$$rj = \max\{\|Sxj,g- Sxj,i\|\} \quad (10)$$

This radius is used for adding particles to a sub-swarm and to merge two sub-swarms. If a particle has a distance less than rj to the initial particle in a sub-swarm, then the particle is absorbed in the sub-swarm. If the distance between the initial particles of two sub-swarm is less than the sum of its radius, then the sub-swarms are merged. This condition is described in equation (11).

$$\|Sxj1,g- Sxj2,g\| <(rj1 + rj2) \quad (11)$$

If a sub-swarm has converged to an optimum it is possible that its radius rj has a value of zero. In this case, it is

difficult to determine the merge of two sub-swarms. A threshold value μ is given and if the distance between initial particles of sub-swarms with a radius close to zero is less than μ, then the sub-swarms are merged. This is expressed in equation (12).

$$||Sxj1,g- Sxj2,g|| < \mu \quad (12)$$

The NichePSO algorithm does not require knowing ahead of time the number of optima of the function. Also, it does not require setting up a fixed radius for the niches. However, it depends on a minimum threshold to determine when a particle does not show changes and a new niche can be created. It also depends on a minimum distance to determine when two sub-swarms that are close to converging to an optimum can be merged.

Algorithm 4: Pseudo Code of NPSO Algorithm

```
Initialize 1 main particle swarm;
    Train main swarm particles using the cognition only model;
     Update fitness of each main swarm particle;
       For each sub-swarm do
         Train sub-swarm particles using one iteration of the GCPSO algorithm;
           Update each particle's fitness;
             Update swarm radius;
end
If possible, merge sub-swarms;
   Allow sub-swarms to absorb any particles from the main swarm that moved into it;
   Search the main swarm for any particle that meets the partition criteria;
If any is found, then create a new sub-swarm with this particle and its closest
neighbor;
```

4. COMPARISON OF PSO WITH OTHER EVOLUTIONARY METHODS

Angeline in 1998 [21] did an early study to compare the particle swarm approach and evolutionary computation in terms of their performance in solving four nonlinear functions, which have been well-studied in the evolutionary optimization literature. He concluded that the performance of the two methods was competitive. Particularly, PSO often locates the near-optimum significantly faster than by evolutionary optimization but cannot dynamically adjust its velocity step size to continue optimization.

Kennedy and Spears [22] compared the PSO algorithm with three versions of genetic algorithm (GA), without mutation; without crossover; and the standard GA which has a crossover, mutation, and selection, in a factorial time-series experiment. They found that all algorithms improved over time, but the PSO found the global optimum on every trial, under every condition. In short, PSO appears to be robust and shows superiority over all versions of GA in almost every cases.

Hasen et al. [23] examined the effectiveness of PSO in finding the true global optimal solution and made a comparison between PSO and GA in terms of their effectiveness and their computational efficiency by implementing statistical analysis and formal hypothesis testing. The performance comparison of the GA and PSO was implemented using a set of benchmark test problems as well as two problems of space system design optimization, namely, telescope array configuration and spacecraft reliability-based design. They showed that the difference in the computational effort between PSO and the GA was problem dependent. It appears that PSO outperforms GA by a large differential in computational efficiency when used to solve unconstrained nonlinear problems with 22 continuous design variables and with low - efficiency differential when applied to constrained nonlinear problems with continuous or discrete design variables.

Lee et al. [24] implemented PSO and compared it with GA to find technical trading rules in the stock market. It was found that PSO could reach the global optimal value with less iteration and kept equilibrium when compared to GA. Moreover, PSO showed the possibility of solving complicated problems without using the crossover, mutation, and other manipulations as in GA but using only basic equations.

Elbeltagi et al. [25] compared five evolutionary algorithms: GAs, Memetic Algorithms (MAs), PSO, and Shuffled Frog Leaping algorithm (SFL) in solving two benchmark continuous optimization test problems. The PSO method was generally found to perform better than the other algorithms in terms of the success rate and the solution quality while being second best in terms of the processing time.

Allahverdi and Al-anzi[26] conducted extensive computational experiments to compare the three methods: PSO, Tabu search, and Earliest Due Date (EDD) along with a random solution in solving an assembly flow shop scheduling problem. The computational analysis indicated that the PSO significantly outperformed the others for difficult problems.

5. APPLICATION

Antennas Design

The optimal control and design of phased arrays, broadband antenna design and modeling, reflector antennas, design of Yagi-Uda arrays, array failure correction, optimization of a reflectarray antenna, far-field radiation pattern reconstruction, antenna modelling, design of planar antennas, conformal antenna array design, design of patch antennas, design of a periodic antenna arrays, near-field antenna measurements, optimization of profiled corrugated horn antennas, synthesis of antenna arrays, adaptive array antennas, design of implantable antennas.

Signal Processing

Pattern recognition of flatness signal, the design of IIR filters, 2D IIR filters, speech coding, analog filter tuning, particle filter optimization, nonlinear adaptive filters, Costas arrays, wavelets, blind detection, blind source separation, localization of acoustic sources, distributed odor source localization and so on.

Networking

Radar networks, bluetooth networks, auto tuning for universal mobile telecommunication system networks, optimal equipment placement in mobile communication, TCP network control, routing, wavelength division multiplexed network, peer-to-peer networks, bandwidth and channel allocation, WDM telecommunication networks, wireless networks, grouped and delayed broadcasting, bandwidth reservation, transmission network planning, voltage regulation, network reconfiguration and expansion, economic dispatch problem, distributed generation, micro grids, congestion management, cellular neural networks, design of radial basis function networks, feed forward neural network training, product unit networks, neural gas networks, design of recurrent neural networks, wavelet neural networks, neuron controllers, wireless sensor network design, estimation of target position in wireless sensor networks, wireless video sensor networks optimization.

Biomedical

Human tremor analysis for the diagnosis of Parkinson’s disease, inference of gene regulatory networks, human movement biomechanics optimization, RNA secondary structure determination, phylogenetic tree reconstruction, cancer classification, and survival prediction, DNA motif detection, biomarker selection, protein structure prediction and docking,

drug design, radiotherapy planning, analysis of brain magnetoencephalography data, electroencephalogram analysis, biometrics and so on.

Electronics and Electromagnetic

On-chip inductors, configuration of FPGAs and parallel processor arrays, fuel cells, circuit synthesis, FPGA-based temperature control, AC transmission system control, electromagnetic shape design, microwave filters, generic electromagnetic design and optimization applications, CMOS RF wideband amplifier design, linear array antenna synthesis, conductors, RF IC design and optimization, semiconductor optimization, high-speed CMOS, frequency selective surface and absorber design, voltage flicker measurement, shielding, digital circuit design.

Robotics

Control of robotic manipulators and arms, motion planning and control, odour source localization, soccer playing, robot running, robot vision, collective robotic search, transport robots, unsupervised robotic learning, path planning, obstacle avoidance, swarm robotics, unmanned vehicle navigation, environment mapping, voice control of robots, and so forth.

Design and Modelling

Conceptual design, electro magnetics case, induction heating cooker design, VLSI design, power systems, RF circuit synthesis, worst case electronic design, motor design, filter design, antenna design, CMOS wideband amplifier design, logic circuits design, transmission lines, mechanical design, library search, inversion of underwater acoustic models, modeling MIDI music, customer satisfaction models, thermal process system identification, friction models, model selection, ultra wide band channel modeling, identifying ARMAX models, power plants and systems, chaotic time series modeling, model order reduction.

Image and Graphics

Image segmentation, auto cropping for digital photographs, synthetic aperture radar imaging, locating treatment planning landmarks in orthodontic x-ray images, image classification, inversion of ocean color reflectance measurements, image fusion, photo time-stamp recognition, traffic stop-sign detection, defect detection, image registration, microwave imaging, pixel classification, detection of objects, pedestrian detection and tracking, texture synthesis, scene matching, contrast enhancement, 3D recovery with structured beam matrix, character recognition, image noise cancellation.

Power Generation and Controlling

Automatic generation control, power transformer protection, power loss minimization, load forecasting, STATCOM power system, fault-tolerant control of compensators, hybrid power generation systems, optimal power dispatch, power system performance optimization, secondary voltage control, power control and optimization, design of power system stabilizers, operational planning for cogeneration systems, control of photovoltaic systems, large-scale power plant control, analysis of power quality signals, generation planning and restructuring, optimal strategies for electricity production, production costing, operations planning.

Fuzzy Systems, Clustering, Data Mining

Design of neurofuzzy networks, fuzzy rule extraction, fuzzy control, membership functions optimization, fuzzy

modeling, fuzzy classification, design of hierarchical fuzzy systems, fuzzy queue management, clustering, clustering in large spatial databases, document and information clustering, dynamic clustering, cascading classifiers, classification of hierarchical biological data, dimensionality reduction, genetic-programming-based classification, fuzzy clustering, classification threshold optimization, electrical wader sort classification, data mining, feature selection.

Optimization

Electrical motors optimization, optimization of internal combustion engines, optimization of nuclear electric propulsion systems, floor planning, travellingsales man problems, n-queens problem, packing and knapsack, minimum spanning trees, satisfiability, knights cover problem, layout optimization, path optimization, urban planning, FPGA placement, and routing.

Prediction and Forecasting

Water quality prediction and classification, prediction of chaotic systems, stream flow forecast, ecological models, meteorological predictions, prediction of the flow stress in steel, time series prediction, electric load forecasting, battery pack state of charge estimation, predictions of elephant migrations, prediction of surface roughness in end milling, urban traffic flow forecasting and so on.

6. CONCLUSIONS

In this paper, we have explained the pseudo code of various PSO, like VNPSO, NPSO, APSO. We have compared it with PSO and applications of PSO is also explained in details to get the better idea of PSO .

7. REFERENCES

1. *Zhanghui Liu and Xiaoli Wang, "A PSO-Based Algorithm for Load Balancing in Virtual Machines of Cloud Computing Environment," Advances in Swarm Intelligence, vol. 7331, pp. 142-147, 2012.*
2. *Chitra, S., et al, "Local minima jump PSO for workflow scheduling in cloud computing environments," Lecture Notes in Electrical Engineering, pp. 1225- 1234, 2014.*
3. *Mohammad Reza Bonyadi and ZbigniewMichalewicz, " Analysis of Stability, Local Convergence, and Transformation Sensitivity of a Variant of the Particle Swarm Optimization Algorithm & IEEE Transactions on Evolutionary Computation, vol. 20, no. 3, pp. 370-385, 2016.*
4. *Marco A. Montes de Oca, Thomas Stutzle, Mauro Birattari and MarcoDorigo, Frankenstein's PSO: A Composite Particle Swarm Optimization Algorithm, IEEE Transactions on Evolutionary Computation, vol. 13, no. 5, pp. 1120-1132, 2009.*
5. *F. van den Bergh and A. P. Engelbrecht, & A Cooperative approach to particle swarm optimization & IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 225-239, 2004.*
6. *Rahim Saeidi, Hamid Reza SadeghMohammadi, Todor Ganchev and Robert David Rodman, Particle Swarm Optimization for Sorted Adapted Gaussian Mixture Models & IEEE Transactions on Audio, Speech, and Language Processing, vol. 17, no. 2, pp. 344-353, 2009.*
7. *Hongbo Liu, Ajith Abraham, Vaclav Snasel and Sean McLoone, "Swarm scheduling approaches for work-flow applications with security constraints in distributed data-intensive computing environments," Information Sciences, vol. 192, pp. 228-243. June 2012.*

8. *J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," IEEE Trans. Evol. Comput., vol. 10, no. 3, pp. 281–295, Jun. 2006.*
9. *Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in Proc. IEEE World Congr. Comput. Intell., 1998, pp. 69–73.*
10. *Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in Proc. IEEE Congr. Evol. Comput., 1999, pp. 1945–1950.*
11. *A. Chatterjee and P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," Comput. Oper. Res., vol. 33, no. 3, pp. 859–871, Mar. 2004.*
12. *Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in Proc. IEEE Congr. Evol. Comput., 2001, vol. 1, pp. 101–106.*
13. *A. Ratnaweera, S. Halgamuge, and H. Watson, "Particle swarm optimization with self-adaptive acceleration coefficients," in Proc. 1st Int. Conf. Fuzzy Syst. Knowl. Discovery, 2003, pp. 264–268.*
14. *A. Ratnaweera, S. Halgamuge, and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," IEEE Trans. Evol. Comput., vol. 8, no. 3, pp. 240–255, Jun. 2004.*
15. *Chowdhury, Aditya, and Vishnu G. Nair. "On the application of particle swarm optimization in minimum time launch vehicle trajectory optimization problem." International Journal of Mechanical and Production Engineering Research and Development 7.5 (2017): 359-366.*
16. *T. Yamaguchi and K. Yasuda, "Adaptive particle swarm optimization: Self-coordinating mechanism with updating information," in Proc. IEEE Int. Conf. Syst., Man, Cybern., Taipei, Taiwan, Oct. 2006, pp. 2303–2308.*
17. *P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, "Adaptive multi-objective particle swarm optimization algorithm," in Proc. IEEE Congr. Evol. Comput., Singapore, 2007, pp. 2281–2288.*
18. *J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in Proc. IEEE Int. Conf. Neural Netw., Perth, Australia, 1995, vol. 4, pp. 1942–1948.*
19. *R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proc. 6th Int. Symp. Micromachine Human Sci., Nagoya, Japan, 1995, pp. 39–43.*
20. *Brits, R., Engelbrecht, A., van den Bergh, F.: A Niching Particle Swarm Optimizer. In Wang, L., Tan, K.C., Furuhashi, T., Kim, J.H., Yao, X., eds.: Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02). Volume 2., Orchid Country Club, Singapore, Nanyang Technical University (November 2002) 692–696.*
21. *Alsoufi, Mohammad S., and Mohammed Yunus. "Effect of Heat Treatment on Stress Corrosion cracking Resistance of Al-Zn-Mg-Cu Alloy Used in Aerospace Engineering Applications."*
22. *van den Bergh, F., Engelbrecht, A.: A new locally convergent particle swarm optimiser. In: 2002 IEEE International Conference on Systems, Man and Cybernetics. Volume 3., IEEE Press (October 2002)*
23. *Angeline, P. J., "Evolutionary Optimization Versus Paricle Swarm Optimization: Philosophy and Performance Differences," Lecture Notes in Computer Science, pp. 601-610, 1998.*
24. *Kennedy, J., and Spears, W. M., "Matching Algorithms to Problems: an Experimental Test of the Particle Swarm and some Genetic Algorithms on the Multimodal Problem generator," Proceedings of the IEEE International Conference on Evolutionary Computation, Anchorage, AK, pp. 78–83, 1998.*

25. *Hassan, R., Cohanim, B., De Weck, O., "A Comparison of Particle Swarm Optimization and the Genetic Algorithm," Proceedings of the 1st AIAA Multidisciplinary Design Optimization Specialist Conference, Austin, TX, 2005.*

26. *Lee, J., Lee, S., Chang, S., "A Comparison of GA and PSO for Excess Return Evaluation in Stock Markets," Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, pp. 221-230, 2005.*

27. *Elbeltagi, E., Hegazy, T., and Grierson, D., "Comparison among Five Evolutionary- Based Optimization Algorithms," Advanced Engineering Informatics, Vol.19, No.1, pp. 43-53, 2005.*

28. *Allahverdi, A., and Al-Anzi, F. S., "A PSO and a Tabu Search Heuristics for the Assembly Scheduling Problem of the Two-Stage Distributed Database Application," Computers & Operations Research, Vol.33, No.4, pp. 1056-1080, 2006.*

29. *Pierre Hansen, Nenad Mladenovic, "Variable Neighborhood search: Principles and applications "European journal of operational research 130(2001)449-467*

30. *Julio Barrera and Carlos A. CoelloCoello, A Review of Particle Swarm Optimization Methods used for Multimodal Optimization".*