International Journal of Science and Research (IJSR)
ISSN: 2319-7064
ResearchGate Impact Factor (2018): 0.28 | SJIF (2018): 7.426

A Comparative Study of Genetic Algorithm and Particle Swarm Optimization in Context of Plant Layout Optimization

Sharad Kumar Rathore[1], Dr. P. M. Mishra[2]

[1]Research Scholar, Department of Mechanical Engineering, Maulana Azad National Institute of Technology, Bhopal, India

[2]Assistant Professor, Department of Mechanical Engineering, Maulana Azad National Institute of Technology, Bhopal, India

Abstract: ***Facility layout problem, which involves planning, designing and optimization of physical arrangement of resources, has significant impact on manufacturing systems. A good placement of facilities contributes to the overall efficiency of operations and reduces total operating expenses. Because of its importance, the facility layout problem has attracted attention of many researchers. Due to the combinatorial nature of this problem, during the last decades, several metaheuristics have been applied to obtain efficient solutions. These approaches have also provided a new perspective on this area. Nowadays many researches are going on using hybrid algorithms and artificial intelligence techniques to optimize layout problems in which multiobjective functions are taken into considerations. Genetic Algorithms (GA) and Particle swarm optimization (PSO) techniques are very adaptively used to solve multiobjective complex problems. The two approaches find a solution to a given objective function employing different procedures and computational techniques; as a result their performance can be evaluated and compared. This paper attempts to examine the claim that PSO has the same effectiveness as the GA but with significantly better computational efficiency (less function evaluations) by implementing statistical analysis and formal hypothesis testing. This paper proposes a new technique that depends basically on forcing PSO to start from initial solutions that guarantee feasible domain obtained using GA. Thus, PSO will be able to define the global optimal solution avoiding the long processing time associated with GA. The major objective of this paper is to compare the computational effectiveness and efficiency of the GA and PSO using a formal hypothesis testing approach***

Keywords: Facility layout problem, Genetic algorithm (G.A), Metaheuristic, particle Swarm Optimization (PSO)

1. Introduction

The decision of where the facilities will be located and the efficient design of those facilities are important and fundamental strategic issues facing any manufacturing industry. The Genetic Algorithm (GA) was introduced in the mid 1970s by John Holland and his colleagues and students at the University of Michigan.[1] The GA is inspired by the principles of genetics and evolution, and mimics the reproduction behavior observed in biological populations. The GA employs the principal of "survival of the fittest" in its search process to select and generate individuals (design solutions) that are adapted to their environment (design objectives/constraints). Therefore, over a number of generations (iterations), desirable traits (design characteristics) will evolve and remain in the genome composition of the population (set of design solutions Generated each iteration) over traits with weaker undesirable characteristics.

The GA is applied to solve complex design optimization problems because it can handle both discrete and continuous variables and nonlinear objective and constrain functions without requiring gradient information [6].

Particle Swarm Optimization (PSO) was invented by Kennedy and Eberhart in the mid 1990s while attempting to simulate the choreographed, graceful motion of swarms of birds as part of a socio cognitive study investigating the notion of "collective intelligence" in biological populations [2]. In PSO, a set of randomly generated solutions (initial swarm) propagates in the design space towards the optimal solution over a number of iterations (moves) based on large amount of information about the design space that is assimilated and shared by all members of the swarm. PSO is inspired by the ability of flocks of birds, schools of fish, and herds of animals to adapt to their environment, find rich sources of food, and avoid predators by implementing an "information sharing" approaches, hence, developing an evolutionary.

2. Literature Survey

Due to the size and availability, the publications were reviewed in the Scopus database. The database search was focused on the problem of facility layout optimization mentioned in keywords, titles and abstracts. The analysis covered the years 1975–2017. 341 studies were found registered in the Scopus database for the analyzed period. The importance of the topic is constantly growing. This statement is supported by the growing number of studies available in Scopus database from one year to another. The number of publications in particular years of the analyzed period is shown in Fig. 1

Licensed Under Creative Commons Attribution CC BY

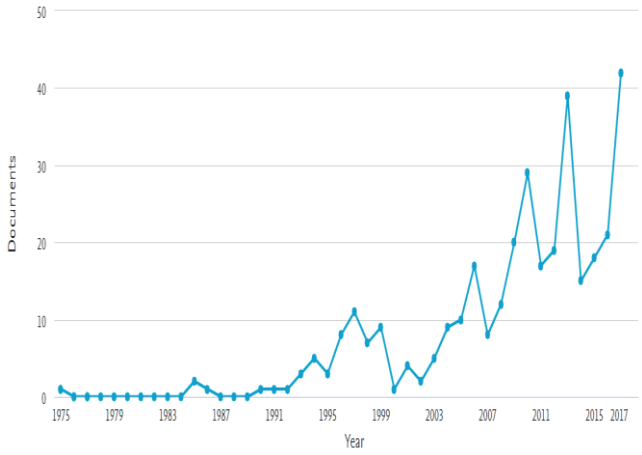


Figure 1: Number of publications indexed in 1975–2017 in Scopus database referring to the facility layout optimization Source: elaborated by the authors based on (http://bazy.pb.edu.pl). Mateusz Kikolski, Chien-Ho Ko international society for manufacturing service and management engineering, vol 10 issue 3 2018 pages 70-79

J. H. Holland [1] presented the various basics of the genetic algorithm and gave a formal setting to the difficult optimization problems characterized by the conjunction of (1) substantial complexity and initial uncertainty, (2) the necessity of acquiring new information rapidly to reduce the uncertainty, and (3) a requirement that the new information be exploited as acquired so that average performance increases at a rate consistent with the rate of acquisition of information.

J. Eberhart et al [2] introduced the concept for the optimization of nonlinear functions using particle swarm methodology. The evolution of several paradigms is outlined, and an implementation of one of the paradigms is discussed. Benchmark testing of the paradigm is described, and applications, including nonlinear function optimization and neural network training, are proposed. The relationships between particle swarm optimization and both artificial life and genetic algorithms are described.

Engelbrecht [3] provided a comprehensive introduction to the new computational paradigm of Swarm Intelligence (SI), a field that emerged from biological research and introduces the various mathematical models of social insects collective behavior, and shows how they can be used in solving optimization problems.

Nadia Nedjah, et al [4] presented some of the most innovative and intriguing applications and additions to the methodology and theory of multi-objective swarm Intelligence -the imitation of social swarm's behaviors for the solution of optimization problems with respect to many criteria.

A Kumar et al [5] demonstrated a comparative study which shows that the HPSO yields improved performance in terms of faster, matured, and accurate localization as compared to global best (gbest) PSO. The performance results on experimental sensor network data demonstrate the effectiveness of the proposed algorithms by comparing the performance in terms of the number of nodes localized, localization accuracy and computation time.

S. Singh et al [6] proposed the application of different migration variants of Biogeography-Based Optimization (BBO) algorithms and Particle Swarm Optimization (PSO) for distributed optimal localization of randomly deployed sensors. An investigation on distributed iterative localization is demonstrated. A comparison of the performance of PSO and different migration variants of BBO in terms of number of nodes localized, localization accuracy and computation time is presented.

O. Maimon et al [7] presented a genetic algorithm approach to the component switching problem. The simplicity and robustness made GA attractive especially when it is combined with modern computing power. This approach can deal with `look-ahead' consideration of component switches, thus transcending the disadvantage of decoupling the PCB sequencing sub-problem from the component loading sub-problem.

3. Genetic Algorithm

In a genetic algorithm (GA), a population of strings (called chromosomes or the genotype of the genome), which encode candidate solutions (called phenotypes) to an optimization problem, evolves toward better solutions. Solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. The algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached. A typical genetic algorithm requires:

- A genetic representation of the solution domain
- **A** fitness function to evaluate the solution domain.

The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size, which facilitates simple crossover operations. Variable length representations may also be used, but crossover implementation is more complex in this case.

3.1 Objective Function of Genetic Algorithm

The principles of GA can be represented in different stages as shown in the Figure 1. The different stages of generational GA are population initialization, selection of individuals for the generation of the new population, and genetic crossover and mutation operations. The algorithm stops as soon as the termination criterion is satisfied such as for example a

Licensed Under Creative Commons Attribution CC BY

maximum number of generations, a detection of convergence of the problem The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. For instance, in the knapsack problem, one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity.

The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid. In some problems, it is hard or even impossible to define the fitness expression; in these cases, interactive genetic algorithms are used.

Once we have the genetic representation and the fitness function defined, GA proceeds to initialize a population of solutions randomly, and then improve it through repetitive application of mutation, crossover, and inversion and selection operators. The genetic algorithm uses the chromosomes fitness value to create a new population consisting of the fittest members. The flow chart of the GA is explained in Fig. 1.

3.1.1 Steps in Genetic Algorithm

The various steps involved in this algorithm are:

- Define an initial population randomly or heuristically.
- Calculate the fitness value for every member inside the population.
- Assign the selection probability for every member in such a way that it is proportional to its fitness value.
- Formulate the next generation from the current generation by selecting the desired individuals to produce off springs.
- Repeat the steps until suitable solution is found.

GA defines a collection of particles known as population and each individual particle is called as chromosome. These chromosomes are then evaluated using the cost function also known as the fitness function. The cost function is usually the objective function of the given problem.

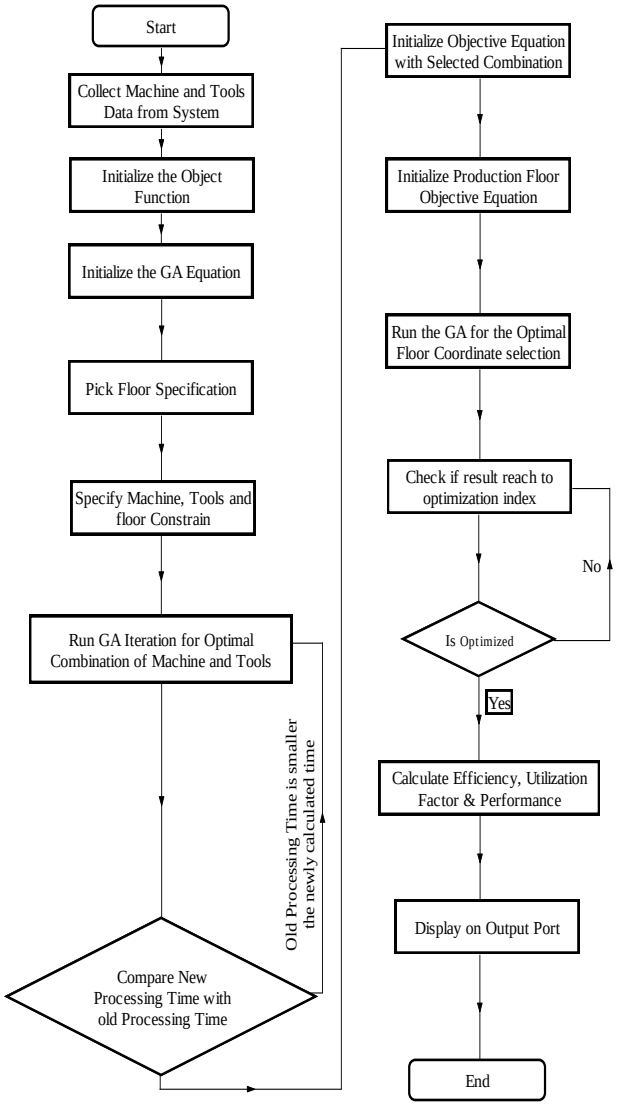


Figure 1: Flow Chart of Genetic Algorithm

The various processes associated with GA are:

a) **Selection** – this process is generally used to choose the c chromosome which will go on to reproduce based on the fitness criterion.
b) **Reproduction** – this step is used for the formation of next generation from the current one.
c) **Crossover** – this process is used to exchange genetic material between the chromosomes. Single or multipoint crossover can be used.
d) **Mutation** – this process leads to the change in chromosomes for a single individual. Mutation prevents the algorithm from getting stuck at a particular point.
e) Stopping criteria – this is the final step in GA. The iteration stops when it reaches a desired solution or it achieves the maximum number of cycles.

Implementation Algorithm: The genetic algorithm uses the chromosomes fitness value to create a new population consisting of the fittest members. The flow chart of the GA is depicted in Fig. 1.

3.1.2 Advantages of Genetic Algorithm

Licensed Under Creative Commons Attribution CC BY

Following are the advantages offered by implementation of Genetic Algorithm.

1) Genetic algorithms search parallel from a population of points. Therefore, it has the ability to avoid being trapped in local optimal.
2) Genetic algorithms use probabilistic selection rules, not deterministic ones.
3) Genetic algorithms work on the Chromosome, which is encoded version of potential solutions' parameters, rather the parameters themselves.
4) Genetic algorithms use fitness score, which is obtained from objective functions, without other derivative or auxiliary information

3.1.3 Limitations of Genetic Algorithm

Following are the limitations of Genetic Algorithm.

1) Take long time to reach to convergence
2) No guarantee of finding global maxima. But then again, apart from brute force, there is rarely any guarantee
3) Totally depends upon trial and error technique, nothing can be controlled during optimization process
4) Complex Technique
5) Incomprehensible solutions

3.1.4 Applications:

Genetic algorithms find its application in manufacturing, and production industries, bioinformatics, computational science, engineering, economics, and other fields.

4. Particle Swarm Optimization (P.S.O)

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 Such methods are commonly known as metaheuristics as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. PSO does not use the gradient of the problem being optimized, which means PSO does not require for the optimization problem to be differentiable as is required by classic optimization methods such as gradient descent and quasi-Newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc. PSO optimizes 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. The movements of the particles are guided by the best found positions in the search-space which are updated as better positions are found by the particles. PSO algorithm works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then it will guide the movements of the swarm. The process is repeated and satisfactory solution will be discovered.

4.1 PSO Variants

Various variants of a basic PSO algorithm are possible. New and some more sophisticated PSO variants are continually being introduced in an attempt to improve optimization performance. There is a trend in that research; one can make a hybrid optimization method using PSO combined with other optimization techniques [3, 5].

- Discrete PSO
- Constriction Coefficient
- Bare-bones PSO
- Fully informed PSO.

4.2 Applications

The first practical application of PSO was in the field of neural network training and was reported together with the algorithm itself (Kennedy and Eberhart 1995). Many more areas of application have been explored ever since, including telecommunications, control, data mining, design, combinatorial optimization, power systems, signal processing, and many others. PSO algorithms have been developed to solve:

- Constrained optimization problems
- Min-max problems
- Multi objective optimization problems
- Dynamic tracking.

4.3 Implementation Algorithm

The PSO algorithm is simple in concept, easy to implement and computational efficient. Original PSO was implemented in a synchronous manner but improved convergence rate is achieved by asynchronous PSO. PSO algorithm is depicted in Figure 2.

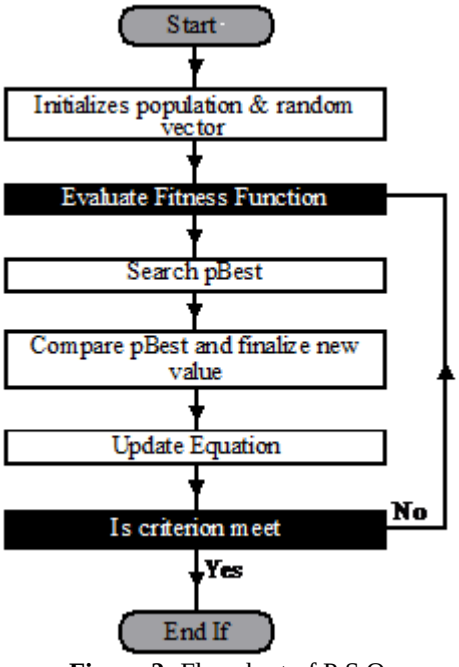


Figure 2: Flowchart of P.S.O

The various steps used in the PSO algorithm are given below:

Licensed Under Creative Commons Attribution CC BY

1) Initialize the particles with some arbitrary velocities and positions in the search space.
2) Start calculating the corresponding value of fitness function of the swarm particles
3) Equate the fitness value evaluation with the current the value of particle's pbest. If current value is better than pbest, set it as new pbest value and set the pbest location to the current location in n-dimensional space;
4) Next equate the fitness value with the previous overall best. If current value is better than gbest, then reset gbest to the current particle's array index and value;
5) Finally assign these values to the corresponding position and velocity of the swarm particle
6) After finding the two best values, the particle updates its velocity and positions with following equation (i) and (ii).

v[ ] = v[ ] + c1 * rand( ) * (pbest[ ] - present[ ]) + c2 * rand( ) * (gbest[ ] – present[ ]) ……..(i)

present[ ] = present [] +v[…..]…… (ii)

v [ ] is the particle velocity, present [ ] is the current particle (solution). **pbest [ ]** and **gbest [ ]** are defined as stated before. **rand ( )** is a random number between (0, 1). c1, c2 are learning factors. Usually c1 = c2 = 2.
The generalized procedure is as follow:-
For each particle

- Initialize particle
- END
- Do
- For each particle
- Calculate fitness value.
- If the fitness value is better than the best fitness value (pbest) in history.
- Set current value as the new pbest
- End.
- Choose the particle with the best fitness value of all the particles as the gbest for each particle
- Calculate particle velocity according equation (a)
- Update particle position according equation (b)
- End while maximum iterations or a minimum error criterion is not attained.

4.4 Advantages of PSO

- Greater diversity and exploration over a single population.
- Momentum effects on particle movement can allow faster convergence
- PSO is a parallel optimization strategy, offer more variety & diversity in trajectories

Both algorithms start with a group of a randomly generated population; both have fitness values to evaluate the population. Both update the population and search for the optimum with random techniques. Both systems do not guarantee success. However, PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory, which is important to the algorithm Compared with genetic algorithms (GAs), the information sharing mechanism in PSO is significantly different. In GAs, chromosomes share information with each other. So the whole population moves like a one group towards an optimal area. In PSO, only gbest (or lbest) gives out the information to others. It is a one -way information sharing mechanism. The evolution only looks for the best solution. Compared with GA, all the particles tend to converge to the best solution quickly even in the local version in most cases.

4.5 Limitations of PSO

- PSO is a continuous technique that is very poorly suited to combinatorial problems.
- PSO is a one-way information sharing mechanism
- PSO does not support GA operator, adversely affect the comprehensiveness of algorithm
- No guarantee for optimal solution
- Tool may collapse under stresses condition
- Algorithm sometime may trap in local minima or local maxima.

4.6 Applications

The first practical application of PSO was in the field of neural network training and was reported together with the algorithm itself (Kennedy and Eberhart 1995). Many more areas of application have been explored ever since, including telecommunications, control, data mining, design, combinatorial optimization, power systems, signal processing, and many others. PSO algorithms have been developed to solve:

- Constrained optimization problems
- Min-max problems
- Multi objective optimization problems
- Design of non linear plant layouts

5. Comparisonof Particle Swarm Optimization & Genetic Algorithm

Particle Swarm Optimization (PSO) is a relatively recent heuristic search method that is based on the idea of collaborative behavior and swarming in biological populations. PSO is similar to the Genetic Algorithm (GA) in the sense that they are both population-based search approaches and that they both depend on information sharing among their population members to enhance their search processes using a combination of deterministic and probabilistic rules. Conversely, the GA is a well established algorithm with many versions and applications although both GA and PSO form an important part of evolutionary optimization algorithms, they do suffer from some disadvantages which limits their usage to only a few problems. The objective of this research paper is to test the hypothesis that states that although PSO and the GA on average yield the same effectiveness (solution quality), PSO is more computationally efficient than the GA PSO is a population based optimization tool, both have fitness values to evaluate the population, both update the population and search for the optimum with random techniques, both systems do not guarantee success. However, unlike GA, PSO

Licensed Under Creative Commons Attribution CC BY

has no evolution operators such as crossover and mutation. In PSO, particles update themselves with internal velocity. They also have memory, which is important to the algorithm. Also, the potential solutions, called particles, are "flown" through the problem space by following the current optimum particles. Compared to GA, the information sharing mechanism in PSO is significantly different. In GAs chromosomes share information with each other. So the whole population moves like a group toward an optimal area. In PSO, only Gbest gives out the information to others. It is a one-way information sharing mechanism. The evolution only looks for the best solution. Compared with GA, all the particles tend to converge to the best solution quickly even in the local version in most cases

6. Conclusion

Particle Swarm Optimization (PSO) is a relatively recent heuristic search method that is based on the idea of collaborative behavior and swarming in biological populations. PSO is similar to the Genetic Algorithm (GA) in the sense that they are both population-based search approaches and that they both depend on information sharing among their population members to enhance their search processes using a combination of deterministic and probabilistic rules. Conversely, the GA is a well established algorithm with many versions and applications although both GA and PSO form an important part of evolutionary optimization algorithms, they do suffer from some disadvantages which limits their usage to only a few problems. The objective of this research paper is to test the hypothesis that states that although PSO and the GA on average yield the same effectiveness (solution quality), PSO is more computationally efficient than the GA PSO is a population based optimization tool, both have fitness values to evaluate the population, both update the population and search for the optimum with random techniques, both systems do not guarantee success. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, particles update themselves with internal velocity. They also have memory, which is important to the algorithm. Also, the potential solutions, called particles, are "flown" through the problem space by following the current optimum particles. Compared to GA, the information sharing mechanism in PSO is significantly different. In GAs chromosomes share information with each other. So the whole population moves like a group toward an optimal area

7. Scope of Future Work

GA is very helpful when the developer does not have precise domain expertise, because GAs possesses the ability to explore and learn from their domain. PSO can be applied to multi-objective problems, in which the fitness comparison takes into account when moving the PSO particles and non-dominated solutions are stored so as to approximation. In order to overcome these problems a combination of both GA and PSO can be used to improve the overall performance. Blending these two algorithms together means to create a compound algorithm that has practical value and combines the advantages of PSO and GA. So, a hybrid algorithm of GA and PSO can be used for future research work.

References

[1] J. H. Holland, ―Genetic algorithms and the optimal allocation of trials, SIAM Journal on Computing, vol. 2, no. 2, pp. 88–105, 1973.
[2] R. Kennedy, J. Eberhart, ―Particle swarm optimization, Neural Networks, 1995. Proceedings., IEEE International Conference on Neural Networks, Perth, WA, Australia, vol. 4, pp. 1942– 1948, 1995.
[3] A. Engelbrecht, ―Fundamentals of computational swarm intelligence, 2006, Hoboken: John Wiley & Sons, Ltd
[4] S. Singh, S. Shivangna, and E. Mittal, ―Range based wireless sensor node localization using pso and bbo and its variants, in Communication Systems and Network Technologies (CSNT), 2013International Conference on. IEEE, 2013, pp. 309–315.
[5] S.Shabir and Dr.R.Singla-A comparative study of Genetic Algorithm and the Particle Swarm Optimization. IJEE Vol 9 Number 2,(2016) PP.215-223.
[6] A. Kumar, A. Khosla, J. S. Saini, and S. Singh, Meta-heuristic range based node localization algorithm for wireless sensor networks, in Localization and GNSS (ICLGNSS), 2012 International Conference on. IEEE, 2012, pp. 1–7.
[7] M. Clerc and J. Kennedy, ―The particle swarm-explosion, stability, and convergence in a multidimensional complex space, Evolutionary Computation, IEEE Transactions on, vol. 6, no. 1, pp. 58–73, 2002. R. Mendes, ―Population topologies and their influence in particle swarm performance, Ph.D. dissertation, Citeseer, 2004.
[8] T.P.Hong and G.N.Shiu, "Allocating multiple base stations under general power consumption by the particle swarm optimization," in Proc. IEEE Swarm Intell. Symp., 2007, pp. 23–28
[9] K. S. Low, H. A. Nguyen, and H. Guo, "A particle swarm optimization approach for the localization of a wireless sensors network," in Proc. IEEE Int. Symp. Ind. Electron., 2008, pp. 1820–1825
[10] R. V. Kulkarni, G. K. Venayaga moorthy, and M. X. Cheng, "Bio-inspired node localization in wireless sensor networks," in Proc. IEEE Int. Conf. Syst., Man Cybern., San Antonio, TX, Oct. 2009, pp. 205–210
[11] Kung Jeng Wang, Shih-Min Wang, "A Negotiation-Based Capacity-Planning Model "IEEE Transactions on Systems, Man, and Cybernetics, Part C-Applications and Reviews, 2012, Volume: 42, Issue: 6, pp983-993
[12] Hamid Khayyam, Minoo Naebe, Bronwyn Fox, "Dynamic Prediction Models and Optimization of Polyacrylonitrile (PAN) Stabilization Processes for Production of Carbon Fiber", IEEE Journals & Magazines, IEEE Transactions on Industrial Informatics, Year: 2015, Volume: 11, Issue: 4, pp:887-896

Licensed Under Creative Commons Attribution CC BY

[13] Shengping Yu, Ying Tang, "An Effective Heuristic Rescheduling Method for Steelmaking and Continuous Casting Production Process With Multi refining Modes", IEEE Transactions on Systems, Man, and Cybernetics: Systems Year: 2016, Volume: 46, Issue: 12, pp: 1675-1688
[14] Marjorie Kitie Nakano, Ricardo de Almeida, "Automotive Industry Line Board Optimization Through Operations Research Techniques" IEEE Latin America Transactions, Year: 2018, Volume: 16, Issue: 2, pp: 585-591
[15] Tamara Borreguero, Raul Pulido, Alvaro Garcia, "Flexible Job Shop Scheduling With Operators in Aeronautical Manufacturing: A Case Study", IEEE Journals & Magazines, IEEE Access, Year: 2018, Volume: 6, pp: 224-233
[16] Yong Chen, Jionghua Jin "Quality-oriented-maintenance for multiple interactive tooling components in discrete manufacturing processes", IEEE Transactions on Reliability, IEEE Journals & Magazines, Year: 2006, Volume: 55, Issue:1, pp:123-134
[17] Hamid Khayyam, Minoo Naebe, Bronwyn Fox, "Dynamic Prediction Models and Optimization of Polyacrylonitrile (PAN) Stabilization Processes for Production of Carbon Fiber", IEEE Journals & Magazines, IEEE Transactions on Industrial Informatics, Year: 2015, Volume: 11, Issue: 4, pp:887-896
[18] Penghui Liu, JingLiu- China, "Multi-leader PSO (MLPSO): A new PSO variant for solving global optimization problems", Elsevier-Applied Soft Computing, Volume 61, 2017, pp:256-263
[19] Xinye Wu, Jianbo Zhao, Yifei Tong, "Big Data Analysis and Scheduling Optimization System Oriented Assembly Process for Complex Equipment", IEEE Access, 2018 , Volume: 6, pp:36479 – 36486
[20] Yusuke Tominaga, Yoshifumi Okamoto, Shinji Wakao, Shuji Sato, "Binary-Based Topology Optimization of Magnetostatic Shielding by a Hybrid Evolutionary Algorithm Combining Genetic Algorithm and Extended Compact Genetic Algorithm", IEEE Transactions on Magnetics, 2013 , Volume: 49 , Issue: 5, pp: 2093 – 2096
[21] Hui Wei, Xue-Song Tang, "A Genetic-Algorithm-Based Explicit Description of Object Contour and its Ability to Facilitate Recognition", IEEE Transactions on Cybernetics, 2015, Volume: 45, Issue: 11, pp: 2558 – 2571
[22] You-Feng Cheng, Wei Shao, Sheng-Jun Zhang, Ya-Peng Li, "An Improved Multi-Objective Genetic Algorithm for Large Planar Array Thinning", IEEE Transactions on Magnetics, 2016, Volume: 52 , Issue: 3, ASN:9400304
[23] Ryota Kamoshida, "Concurrent optimization of job shop scheduling and dynamic and flexible facility layout planning", 2018 5th International Conference on Industrial Engineering and Applications (ICIEA), 2018, 289 – 293
[24] Tao Zhang, Qi-qi Zhang, Yue-jie Zhang, "Multi-level inventory matching and order planning for steel plants based MTO-MTS", Proceeding of the 11th World Congress on Intelligent Control and Automation, 2014, pp:2735 – 2738
[25] G. Y. Chen, K. J. Rogers, "Proposition of two multiple criteria models applied to Dynamic Multi-objective Facility Layout Problem based on Ant Colony Optimization", 2009 IEEE International Conference on Industrial Engineering and Engineering Management, 2009, pp:1553 – 1557
[26] Inna Khavina, " Multiagent system for the optimal control of a combination of transport robots in an manufacturing technological process", 2016 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2016, pp: 468 – 472
[27] Inna Khavina, "Multiagent system for the optimal control of a combination of transport robots in an manufacturing technological process", 2016 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2016, pp:468 – 472.

Licensed Under Creative Commons Attribution CC BY