International Journal of Electrical Engineering.
ISSN 0974-2158 Volume 9, Number 2 (2016), pp. 215-223
© International Research Publication House
http://www.irphouse.com

A Comparative Study of Genetic Algorithm and the Particle Swarm Optimization

Shahid Shabir
M.Tech. Research Scholar, Department of Electronics and Communication Engg., Chandigarh Engineering College, Landran, Mohali, Punjab, India.

Dr. Ruchi Singla
Professor and HoD, Department of Electronics and Communication Engineering, Chandigarh Engineering College, Landran, Mohali, Punjab, India.

Abstract

Evolutionary algorithms have gained much attention of the researchers as an effective methods for solving different optimization problems. The Genetic Algorithm (GA) is very popular in various fields mainly because of its sense, implementation, and the ability to solve complex problems usually found in engineering systems. The drawback of the GA is that it has high implementation cost and usually requires a higher number of iterations. Particle Swarm Optimization (PSO) is a relatively recent heuristic algorithm which is based on the behavior of swarming characteristics of living organisms. PSO is quite similar to the GA as these two are evolutionary search methods which means that PSO and the GA change from a set of points to another set of points within an iteration with visible improvement from the previous values using some probabilistic and deterministic rules. This paper is used to study the implementation, features and effectiveness of these two evolutionary algorithms.

Keywords: Genetic algorithm (GA), Numerical optimization, Particle Swarm Optimization (PSO), Stochastic, Swarm .

I. INTRODUCTION

Most optimization problems can be solved by using any evolutionary algorithm. One of the most important class of Evolutionary algorithms is Genetic algorithm (GA). The concept of GA was introduced by John Holland in 1970s at University of Michigan [1].Genetic algorithm are categorized as global search heuristics that uses iterative process to obtain desired solutions. GA usually provides approximate

solutions to the various problems. GA uses various biological techniques such as inheritance, selection, crossover or recombination, mutation and reproduction. Since GA is able to handle both discrete and continuous variables, it can be used to solve complex optimization problems.GA has been very efficient in various problems such as optimization, design and scheduling [7], power systems [8],data handling etc.

The optimization problems can also be easily solved by an innovative distributed paradigm known as Swarm Intelligence (SI). The concept of SI was introduced by Gerardo Beni and Jing Wang in 1989, who originally got inspired from the biological examples such as bird flocking, ant colonies, animal herding, fish schooling and bacterial growth. An attempt was made to design various algorithms or distributed problem solving devices based on the biological phenomena or systems. Particle Swarm Optimization (PSO) was developed by Kennedy and Eberhart in the mid 1990s [2]. The fundamental idea in PSO is that each particle represents a potential solution which it updates according to two important kinds of information available in decision process. The first one (cognitive behaviour) is gained by its own experience, and the second one (social behaviour) is the experience gained from the neighbours, that is, they tried the choices itself and have the knowledge which choices their neighbours have outstand so far and how positive the best pattern of choices was. PSO has been used increasingly due to its several advantages like robustness, efficiency and simplicity. When compared with other stochastic algorithms it has been found that PSO requires less computational effort [3] [4]. Although PSO has shown its potential on many aspects for solving different optimization problems, it still requires considerable execution time to find solutions for large-scale engineering problems [5][6].

II. RELATED WORK

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.

A. 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 behaviour, 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 swarms 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.

S. Singh et al [12] present different optimization algorithm with tremendous speedups in the computation time. The overall GPU performance of multi-GPU Island-based GA for solving Knapsack problem reaches5.67 TFLOPS. MINLP archived an overall speedup of 20x to 42x using nVidia Tesla C2050 GPU as compared to Intel Core i7 920 CPU processor. On implementing Steady state GA on a GPU approximately 6 times faster results are obtained than the corresponding CPU implementation.

III. GENETIC ALGORITHM

Genetic Algorithm (GA) is an important class of evolutionary algorithm. In 1975, the genetic algorithm was first of all used by Prof. John Holland (Holland, 1975) [1]. GA usually provides approximate solutions to the various problems. GA uses various biological techniques such as inheritance, selection, crossover or recombination, mutation and reproduction. Real-coded GA is usually faster than binary GA because it does not need binary encoding and decoding. The various steps involved in this algorithm are:

1) Define an initial population randomly or heuristically.
2) Calculate the fitness value for every member inside the population.
3) Assign the selection probability for every member in such a way that it is proportional to its fitness value.
4) Formulate the next generation from the current generation by selecting the desired individuals to produce off springs.
5) Repeat the steps until suitable solution is found.
6) 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. Some of the processes associated with GA are:

a) **Selection** – this process is generally used to choose the 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: GA results in the formation of fittest members after every iteration by using a predefined fitness function. Figure 1 ahead shows the basic flow chart of the Genetic Algorithm.

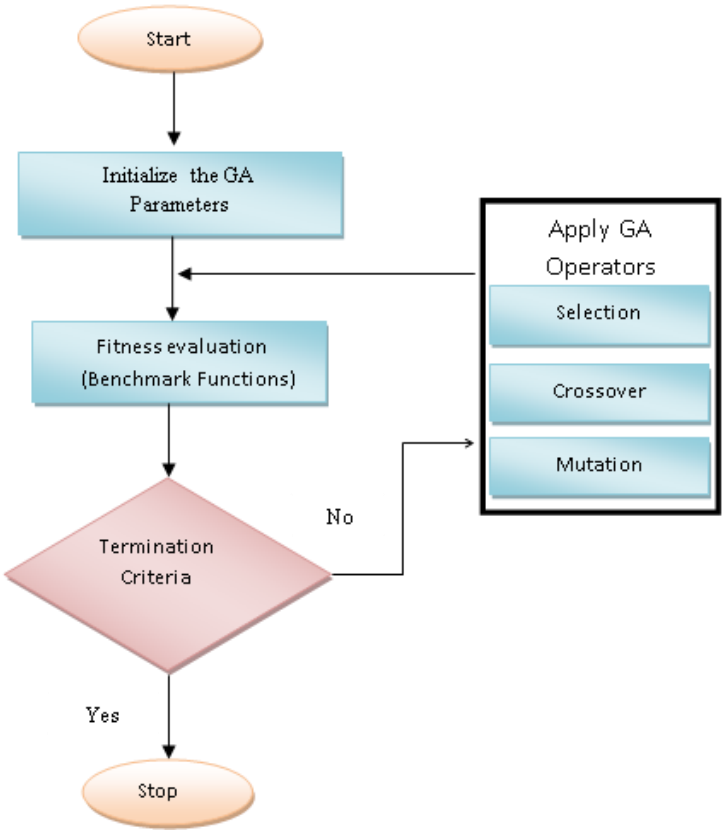

Figure 1: Basic implementation of GA

Applications: Genetic algorithms can be used in a wide variety of fields. It is mainly used to solve optimization problems. Some of the fields where GA is used are: bioinformatics, computational science, electrical engineering, manufacturing, and phylo-genetics, etc

IV. PARTICLE SWARM OPTIMIZATION

PSO is an important class of evolutionary algorithm which defines a swarm of particles. The particles in the swarm are then changed according to the predefined rules. The change in the particle values is determined by their previous position and the best known position of the particle over the entire search space [9] [10].The particles are initialized by a randomized position at the beginning of the search process, and then after every iteration, the position and velocity of each particle is changed in such a way that it moves towards the desired pbest and gbest location. The efficiency of local search and convergence to the global optimum solution are obtained by weighting the acceleration coefficients with random terms [11]. Both pbest and gbest locations generate separate random numbers for acceleration [12].
Consider the n-dimensional search space and let the k-th particle in the swarm is represented by

$X_k = (x_{k1}, x_{k2},.....x_{kd})$ and let its velocity be represented by another n-dimensional vector $V_k = (v_{k1}, v_{k2}, ... v_{kd})$. Let the best position visited by the k-th particle be denoted by $P_k = (p_{k1}, p_{k2}, ... p_{kd})$. The personal best particle is denoted as $P_p=(p_{p1}, p_{p2}, ... P_{pd})$, and the overall best particle be denoted as $P_g = (p_{g1}, p_{g2},..... p_{gd})$, where g and p are particle indices. The position and velocity of the particle can be updated by the given formula below:

$$X_{kd}(t+1) = X_{kd}(t) + V_{kd}(t) \quad .....(1) \quad \text{And}$$

$$V_{kd}(t+1) = \chi\,(v_{kd}(t) + l_1c_1\,(P_{pbd}(t) - X_{kd}(t)) + l_2c_2\,(P_{gbd}(t) - X_{kd}(t))) \quad ....(2)$$

In the above equation l_1 and l_2 are non-negative constants known as the learning factors. Also, c_1 and c_2 are some random numbers generated in the range [0,1]. χ denotes the constriction factor, which is defined as:

$$\chi = \frac{2}{|2-\emptyset-\sqrt{\emptyset^2-4\emptyset}|}$$

Where, $\emptyset = l_1+l_2$

If ψ is set to 4.1, then $\chi = 0.729$ [13][14]. The particle velocity $V_{kd} \in [-V_{min}, V_{max}]$, where Vmax is the maximum velocity. If the velocity exceeds Vmax in any coordinate it will be truncated to Vmax to avoid search explosion. If it is too high, the particles could skip over good solutions and if too small, particles are explored too slowly and good solutions could not be found. Ppbd (t) and Pgbd (t) are the personal and global best position respectively.

Implementation of PSO: PSO is an evolutionary algorithm which requires the generation of random numbers. The performance of PSO algorithm is affected by the quantity and the quality of the numbers generated. The initial iteration is performed over the entire search space. The basic implementation of PSO is shown in the figure 2.

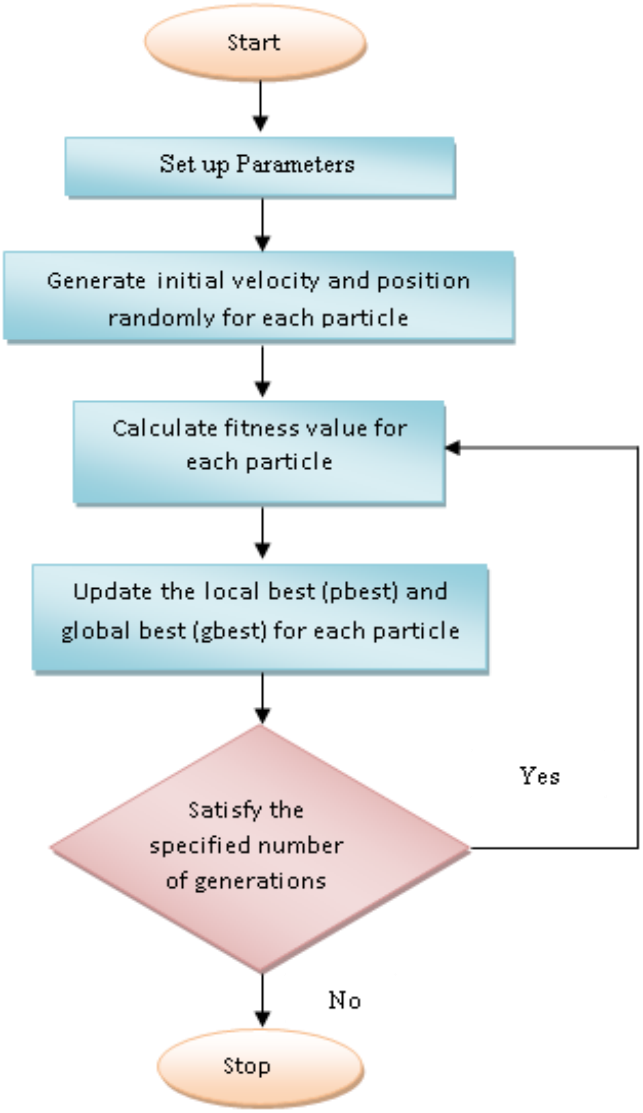

Figure 2: Basic implementation of Particle Swarm Optimization algorithm.

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

i) Initialize the particles with some arbitrary velocities and positions in the search space.

ii) Start calculating the corresponding value of fitness function of the swarm particles.

iii) 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;
iv) 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;
v) Finally assign these values to the corresponding position and velocity of the swarm particle.

PSO Variants: Various versions of PSO algorithm can be obtained by combining it with other evolutionary algorithms. There is a trend in research to make hybrid PSO algorithms to improve the overall optimization of the algorithm. Some commonly used variants of PSO algorithm are: [10].

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

Applications: PSO found its first application in the field of neural network training. Since then it has been used in wide variety of fields including telecommunications, design, power systems, control and many others. PSO algorithms have been used extensively in dynamic tracking, MinMax problems and various optimization problems.

V. GENETIC ALGORITHM VERSUS PARTICLE SWARM OPTIMIZATION

GA is discrete in nature, i.e. it changes the variables into binary 0's and 1's, and therefore it can easily handle discrete problems, and PSO is continuous and hence must be modified in order to handle discrete problems.

Unlike GA, the variables in PSO can take any values based on their current position in the particle space and the corresponding velocity vector.

Genetic algorithms do not handle complexity in an efficient way, because in such cases the number of elements undergoing mutation in very large which causes a considerable increase in the search space. So, in this case PSO is the best alternative as it requires small number of parameters and correspondingly lower number of iterations.

GA usually converges towards a local optimum or even arbitrary points rather than the global optimum of the problem while as PSO tries to find the global optima.

VI. CONCLUSION AND SCOPE OF FUTURE WORK

Particle Swarm Optimization (PSO) is a relatively recent heuristic algorithm which is based on the behavior of swarming characteristics of living organisms. PSO is quite similar to the GA as these two are evolutionary search methods which means that PSO and the GA change from a set of points to another set of points within an

iteration with visible improvement from the previous values using some probabilistic and deterministic rules. Conversely, the GA is a well-established and popular algorithm with many applications and different versions.

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. 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 is a good topic for future research.

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] N. Nedjah, L. dos Santos Coelho, and L. de Macedo Mourelle, ―Multi-objective swarm intelligent systems: theory & experiences.‖ Springer Science & Business Media, 2009, vol. 261.

[5] 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.

[6] 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.

[7] O. Maimon and D. Braha, ―A genetic algorithm approach to scheduling pcbs on a single machine,‖ International Journal of Production Research, vol. 36, no. 3, pp. 761–784, 1998.

[8] J. Zhang, H. Chung, and W.-L. Lo, ―Pseudocoevolutionary genetic algorithms for power electronic circuits optimization,‖ Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 36, no. 4, pp. 590– 598, 2006.

[9] J. Kennedy and R. Mendes, ―Population structure and particle swarm performance,‖ 2002. [10]J. Kennedy, J. F. Kennedy, and R. C. Eberhart, Swarm intelligence. Morgan Kaufmann, 2001.

[11] A. Ratnaweera, S. Halgamuge, and H. C. Watson, ―Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients,‖ Evolutionary Computation, IEEE Transactions on, vol. 8, no. 3, pp. 240–255, 2004.

[12] S. Singh, J. Kaur, and R. S. Sinha, ―A comprehensive survey on various evolutionary algorithms on gpu,‖ 2014.

[13] 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.

[14] R. Mendes, ―Population topologies and their influence in particle swarm performance,‖ Ph.D. dissertation, Citeseer, 2004.

[15] M. Jamil and X.-S. Yang, ―A literature survey of benchmark functions for global optimization problems, int,‖ Journal of Mathematical Modelling and Numerical Optimisation, vol. 4, no. 2, pp. 150–194.