Genetic Algorithm, Particle Swarm Optimization and Harmony Search: A Quick Comparison

Sonia Sharma
Department of Computer Science and Engineering
Amity University Uttar Pradesh
Noida, India
sonia29bhardwaj@gmail.com

Hari Mohan Pandey
Department of Computer Science and Engineering
Amity University Uttar Pradesh
Noida, India
profharimohanpandey@gmail.com, hmpandey@amity.edu

***Abstract*—There exists several complex optimization problems, are difficult to solve using simple conventional or mathematical approach. Many scientific applications have a search space exponentially proportional to the problem dimensions, cannot be solved employing exhaustive search methods. Therefore, there is considerable interest in meta-heuristic methods attempt to discover near optimal solution within the acceptable time. This paper presents a comprehensive study and comparison of three: Genetic Algorithm, Particle Swarm Optimization and Harmony Search, global optimization algorithms. The comparative analysis has been reported in an organized manner for quick review. The underlying motivation is to identify possibility to develop a new hybrid algorithm to solve real world problems.**

Keywords—algorithm, artificial intelligence, meta-heuristic algorithms, optimization, genetic algorithm, particle swarm optimization, harmony search

I. INTRODUCTION

Fred Glover coined the term Metaheuristic. Meta-in an upper level, heuristic-to find, these two combined word algorithms give a high level, best quality, cost efficient, quick and reliable search solution for an optimization problem. Optimization problems are those which are discrete with incomplete information and have limited or weak in computation. Meta-heuristic algorithms considers three factors:

a) An objective function is maximized or minimize;
b) A set of unknowns or variables affects the objective function; and
c) A set of constraints allow the unknown to accept certain values and exclude others, on which optimization problems are centralized [2].

The solutions of the meta-heuristic algorithms are dependent on the set of random variables. The metaheuristic algorithms do not give guarantee of finding the global optimum for some class of problem, but always try to explore all possible regions of the search space, greatly increases the chances of getting nearer optimal solution. The metaheuristic algorithms attracted attention of the researcher and scientists of different disciplines includes physics, chemistry, molecular biology, engineering and economics due the power it poses.

In this paper, we introduced some naturally inspired meta heuristic algorithms, namely Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Harmonic Search (HS) to solve complex optimization problem by randomly continuous search in a large search space. The interesting thing about the selected algorithms are they belongs to different categories of population based metaheuristic algorithms like GA is belongs to evolutionary algorithms (EA), PSO is swarm intelligence (SI) based algorithm, whilst HS is nature inspired algorithm. The comparison of these algorithms is really dramatically interesting thing to do. The motivation of conducting this study is to investigate new possibilities to develop a hybrid algorithm incorporating the features of EA, SI and nature inspired algorithms.

The rest of the paper is organized as follows: Section II discusses the basics of the GA, PSO and HS. The section III presents the role of the key factors affects the working of these algorithms. The comparison of the considered algorithms has also been represented in Section III, while the conclusion has been drawn in the Section IV.

II. META-HEURISTIC ALGORITHMS: GA, PSO AND HS

A. The Genetic Algorithm

The basic concept of the GA was first pioneered by J.H. Holland in 1960. The GA is a mature approach for solving complex and conflicting optimization problems and the optima is obtained, is evolved generation to generation without rigorous mathematical formulation. The solution obtained is the best one forward from previous generation's stronger candidate attributes carried out to current generations. The GA is a nature inspired, biological evolution process in which stronger candidate will be won in a competitive environment and selected. The GA works randomly with encoding of variant of search space into a finite length of the binary string. The string has a stochastic candidate solution for search problem, biologically known as chromosomes and binary bits are known as genes. For examples, we consider a popular travelling salesman problem (TSP) in which the routes treat as chromosome and city treats as genes. The GA has been successfully applied in different areas includes engineering optimization problems, grammar induction [2] [3] [4] [6]. The premature convergence is the key issue with the GA have been discussed in a comprehensive manner in [5]. Following are the pseudocode steps applied in the GA:

Step 1: Initialization

Initialize the population and generate randomly a high quality population, so that an ultimate fittest candidate will be getting.

Step 2: Evolution

Evolve the population to calculate the fitness of each individual solution. There are following steps in evolution process as:

- *Selection*: In this process selects two candidates or chromosome from a population have good fitness and consider them as a parent. Many selection procedures are available for this process like the ranking selection.
- *Reproduction*: In this process crossover is applied on two selected fitted chromosomes from the population that produce a one or two offspring in which best fitted offspring is selected and result install back into the population and remaining least fit population destroy.
- *Crossover*: This process recombines the parts of the parent to produce new offspring which is not identical with its parent.
- *Mutation*: This process makes a change in offspring to maintain the genetic diversity of populations from one generation to the next.

Step 3: Replacement

Replace the new generated offspring population from the parental population and go ahead to run the algorithm.

Step 4: Test

If the final condition is ok then stop and return the best fit designated candidate or chromosome found as far as a solution.

Step 5: Loop Repeat step 2 to 5 until terminated condition is met.

B. The Particle Swarm Optimization

The PSO is a heuristic globally accepted optimization principle was proposed by Kennedy and Eberhert in 1955 [7]. It is inspired from the SI and procedure based on the biological species social behavior like swarming, flocking, herding birds and fish schooling. For example, the fish (candidate or particle) take a suitable path to go (e.g. For food), the rest of the swarm will be able to follow promptly, even if they are on opposite sides, if searching fish have closest source of food (potential solution), they don't have any leader in their group or swarm. The swarm gets best condition simultaneously through communication among their members who already have a better situation. A particle, which has a better condition shares it with the swarm and the others will move simultaneously to that place. This would happen repeatedly until the best conditions or a food source is discovered [7] [8] [9] [10] [11]. In the PSO, particle represents potential solution. With their exploration and exploitation particle of swarm travel through n-dimensional search space. Hypothesis are plotted in search space and seeded with initial velocity with regard of their position. The particles have the best position in its respect is called local-best (*lbest*) and particle has the best position with respect to knowledge of their neighbor and global is called global-best(*gbest*).

Let us consider $x_{id}(t)$ denote the position of i^{th} particle at d-dimension in the search space at time t. After finding optimum value, the particle update its velocity and position as follow:

$$V_{id}(t)=V_{id}(t)+c_1*r(t)*(lbest(t)-x_{id}(t))+c_2*r(t)*gbest(t)-x_{id}(t)) \quad (1)$$

$$x_{id}(t) = x_{id}(t) + V_{id}(t) \quad (2)$$

Where, $V_{id}(t)$ denote velocity of i^{th} a particle with d-dimension at the time stamp t, $r(t)$ denote the random number between $(0,1)$, c_1, and c_2 are learning factor usually $c_1 = c_2 = 2$.

These two constraint apply in the algorithm to control the location and velocity of the particle from the target so that a particle is not too far from the best solution and too small that sudden reach to target and beyond the best solution.

Pseudocode for PSO Algorithm:

Step1: For each particle "Initialize particle" and End

Step 2: Do

For each particle

Calculate the fitness value

If the fitness value is better than its local best

Set current value as the new *lbest*

End

Step 3: Choose the particle with the best fitness value of all as *gbest*

For each particle

Calculate the particle velocity according equation (1)

Update particle position according equation (2)

End

Step 4: While maximum iterations or minimum error criteria is not attained Inertia weight

The basic PSO has some drawbacks, for example, it is partial optimism, not work for problem of scattering and optimization and the problem for non-coordinate system. So to resolve these drawbacks, some modifications have been done in the basic PSO incorporating inertia weight w in the equation (1) as represented in equation (3).

$$V_{id}(t) = w*V_{id}(t) + c_1*r(t)*(lbest(t)-x_{id}(t)) + c_2*r(t)*gbest(t)-x_{id}(t)) \quad (3)$$

Inertia weight is used, controls the current velocity on the base of previous knowledge the velocity. Generally, w is equal to 1. If $w>1$, then the velocity will decrease with time, the particle will accelerate to maximum velocity and the swarm will be divergent. If $w<1$, then the velocity of particle will decrease until it reaches zero. The larger value of w will facilitates an exploration, rather small values will promote the exploitation [10] [11]. As w is decreasing, the velocity of the particle will also get slower down to search for the delicate partical. For complex problem, PSO's searching ability for the whole has not been found effective, the most optimist solution cannot be found, so the inertia weights can be used to work out the problem [11].

C. Harmony Search

The HS is a heuristic approach based on the improvisation process of jazz musicians first introduced by Zong Woo Geem et al. in 2001 [12]. In jazz music different musicians try to find perfect pleasing harmony which determined by audio aesthetic objectives [13] [14] [15] [16]. Similarly like a musician always tries to produce a perfect harmony, an optimal solution to an optimization problem should be the best under given objective and constraints applied to it. The HS has been applied to solve many optimization problems such as an optimization, ground water modeling, vehicle routing and energy-saving dispatch and others. When a musician does improvisation, he has three choices:

a) Playing any one pitch from his (or her) memory;
b) Playing an adjacent pitch of one pitch from his (or her) memory, and
c) Playing totally random pitch from the possible range of pitches.

The three key parameters of harmony have been considered for improvisation are:

- Memory : Harmony memory (HM) will be ensured about the best solution to be carried out over new harmony by random selection from harmony memory size. It is denoted as "r" its value belong to (0,1).
- Pitch(frequency) Adjustment: pitch adjustment parameter is determined by pitch bandwidth "bw" and pitch adjustment rate (PAR)
- Randomization : It is to increase the diversity of the solutions. Although pitch adjustment parameter has a similar role, but it is limited to certain local pitch adjustment and thus corresponds to a local search. The use of randomization can drive the system further to explore various diverse solutions so as to find the global optimality.

According to these three principle rules, HS algorithm is can be defined in 5 steps as follow:

Step 1: Initializing the parameter

{HMS: harmony memory size tells about number of solution vector or population, HMCR: harmony memory considering rate, PAR : pitch adjusting rate, bw: distance bandwidth, $r(0,1)$: uniformly distributed random number between 0 and 1}

Consider the objective function $f(x)$, which is subject to maximize or minimize according to the value of x is a solution vector composed of decision variables x_i, Where $x \in X_i$ and $i = 1,2,3,....N$ and X_i is the range of value for decision variable x_i, and X_i lies between lower bound and upper bound for each variable such as $LB(x_i) \le X_i \le UB(x_i)$, N is number of decision variables.

Step 2 : Initializing the Harmony Memory

Each component says x_i^j in HM of size HMS is initialized with a random number between upper and lower bound, where $1 \le i \le N$. The equation for i^{th} component of the j^{th} solution vector is as follows:

$$x_i^j = LB(x_i) + r(0,1) * (UB(x_i) - LB(x_i)) \quad (4)$$

Where $j = 1,2,3....$ HMS and r is instantiated a new value for each component of each vector.

Step 3: Generating or improvising new Harmony

On the base of three improvising rule a new harmony vector is generated which is first analyze on the HMCR parameter and then again examined on PAR parameter such as: $x' = (x_1', x_2', ...x_N')$, the value of i^{th} component with probability HMCR lies between 0 and 1 is rate of selecting one value from the previous store in HM, while (1- HMCR) is the rate of randomly selecting a new value from the possible rang of value.

$$x_i' = \begin{cases} x_i \in \{x_i^1, x_i^2, ..., x_i^{HMS}\} \\ x_i \in X_i \end{cases} \quad (5)$$

Every component obtained by memory consideration, is analysed to determine whether it should be pitch adjusted, is done using equation (6).

$$x_i' = \begin{cases} x_i' \pm r(0,1) * bw & probability \quad PAR \\ x_i' & probability \quad (1 - PAR) \end{cases} \quad (6)$$

Where, $PAR \in (0,1)$

Step 4 : Harmony memory(HM) evolution or update

In this step objective function value is evaluated as if new harmony vector is better than the old worst harmony, then new harmony is included in the HM and the existing old worst harmony is excluded from the HM.

*Step 5: Terminating condition: i*f the maximum number of improvisation is satisfied, then terminate otherwise repeat step 3 and 4.

III. FACTOR EFFECTS META-HEURISTIC ALGORITHMS AND COMPARISON

- The GA is mainly affected by its fundamental operators mutation and crossover, helps in finding the optimum value. The selection operation is also important in the GA greatly contributes to the success of the search, is performed in two places known as survival selection and parent selections. The crossover should be applied to strings that are able to produce better offspring, if weak

chromosomes are taken for crossover will result in the poor offspring and the solution will not optimal in the next generation. On the other side, the mutation affects the diversity. The population size also affects the GA, the size of the population is not too small that have lack of optimum values and too big so that searching will be time consuming.

- In the PSO population size is an important parameter that converges the algorithm and quality of the solution and large population should not be considered because it increase the computation cost. It should be kept around 20-40 particles. The other parameters are velocity, position of a particle generated randomly and should be updated for *pbest* and *gbest*. Inertia weight is initialized with (0,1) and should be updated in an iterative manner as the search grows. The PSO has shown the ability of balance between local and global minima during updating velocity and position by storing values of *pbest* and *gbest* in earlier iterations.
- The HS has two basic parameters are HMCR and PAR, drastically affects the performance. These parameters control the speed of the convergence and component of the solution. The HMCR is used to set the probability of previous information stored in the HM, for example, if HMCR is 0.8 then the probability to choose each component from new solution is 80% and 10% from entire feasible range and then adjust the PAR accordingly.

The Table I shows the summary of the comparison of the algorithms picked for the present study. The primary focus has been given to the parameters used in these algorithm's implementation and the variations that has been proposed so far.

TABLE I

Comparison of the GA, PSO and HS

Optimization Techniques	**Author (s)**	**Variations**	**Parameters**
Genetic Algorithm	J.H Holland	Traditional variant Real coded GA Binary coded GA Improved GA SAWTOOTH GA Differential evaluation LMS	Population size Diversity Mutation Crossover Selection probability Generation gap Stopping criteria
Particle Swarm Algorithm	Kennedy and Eberhert	Basic variant Velocity clamping Constriction Synchronous Asynchronous Modified variant Discrete PSO • Binary PSO • Integer PSO Complex PSO • Dynamic neighborhood PSO • Constrained handling PSO • Multi-objective Optimization Hybrid PSO Adaptive PSO Combinatorial PSO	Number of particles Velocity Position Random number Range of particle Learning factor Inertia weight Local search and Global search Terminating criteria
Harmonic Search	Zong Woo Geem et al.	Basic HS Variant of HS Dynamic algorithm parameters Modeling dependencies between decision variables Hybridization with sequential quadratic programing	Objective function Solution vector Decision variable Harmony memory size Improvisation Distance bandwidth bw Probability of HMCR Probability of PAR Selection criteria Terminating criteria

IV. CONCLUSIONS

In this paper, the authors have illustrated the comparative study of the three metaheuristic algorithms are: GA, PSO and HS. The basic functionality of these algorithms has been discussed and pseudocode for each algorithm has been shown and, is discussed in significant details. The authors have drawn the effect of the different parameters for each algorithm has been taken in this study and presented a comparison in the tabulated form (Table I). The authors have strongly believed that this paper will be a useful resource for the researchers to develop more improved algorithms in the near future, will show a hybrid feature to solve complex real world problems.

REFERENCES

[1] Holland, John H. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press, 1992.

[2] H.M. Pandey, A. Dixit, D. Mehrotra, Genetic algorithms: concepts, issues and a case study of grammar induction, in: Proceedings of the CUBE International Information Technology Conference, ACM, 2012.

[3] N.S. Choubey, H.M. Pandey, M.U. Kharat, Developing genetic algorithm library using Java for CFG induction, Int. J. Adv. Technol. 2 (1) (2011) 117–128.

[4] H.M. Pandey, Context free grammar induction library using Genetic Algorithms,in: 2010 International Conference on Computer and Communication Technology (ICCCT), IEEE, 2010.

[5] H.M. Pandey, A. Choudhary, D. Mehrotra, A comparative review of approaches to prevent premature convergence in GA, in: Applied Soft Computing,2014.

[6] H.M. Pandey, et al., Grammar induction using bit masking oriented genetic algorithm and comparative analysis, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.09.044

[7] Eberhart, Russ C., and James Kennedy. "A new optimizer using particle swarm theory." *Proceedings of the sixth international symposium on micro machine and human science*. Vol. 1. 1995.

[8] Yang, Xin-She. *Nature-inspired metaheuristic algorithms*. Luniver press, 2010.

[9] Cui, Zhihua, and Xingjuan Cai. "Integral particle swarm optimization with dispersed accelerator information." *Fundamenta Informaticae* 95.4 (2009): 427-447.

[10] Cui, Zhihua, et al. "PID-Controlled Particle Swarm Optimization." *Multiple-Valued Logic and Soft Computing* 16.6 (2010): 585-609.

[11] Gandomi, Amir Hossein, et al. "Chaos-enhanced accelerated particle swarm optimization." *Communications in Nonlinear Science and Numerical Simulation*18.2 (2013): 327-340.

[12] Geem, Zong Woo, Joong Hoon Kim, and G. V. Loganathan. "A new heuristic optimization algorithm: harmony search." *Simulation* 76.2 (2001): 60-68.

[13] Weyland, Dennis. "A Rigorous Analysis of the Harmony Search Algorithm: How the Research Community can be." *Modeling, Analysis, and Applications in Metaheuristic Computing: Advancements and Trends: Advancements and Trends* (2012): 72.

[14] Yang, Xin-She. "Harmony search as a metaheuristic algorithm." *Music-inspired harmony search algorithm*. Springer Berlin Heidelberg, 2009. 1-14.

[15] Wang, Ling, et al. "A hybrid binary harmony search algorithm inspired by ant system." *Cybernetics and Intelligent Systems (CIS), 2011 IEEE 5th International Conference on*. IEEE, 2011.

[16] Chakraborty, Prithwish, et al. "An Improved Harmony Search Algorithm with Differential Mutation Operator." *Fundam. Inform.* 95.4 (2009): 401-426.