# A New Hybrid PSOGSA Algorithm for Function Optimization

Seyedali Mirjalili[#1], Siti Zaiton Mohd Hashim[#2]
[#]Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia
Johor Bahru, Malaysia
[1]ali.mirjalili@gmail.com
[2]sitizaiton@utm.my

*Abstract*— **In this paper, a new hybrid population-based algorithm (PSOGSA) is proposed with the combination of Particle Swarm Optimization (PSO) and Gravitational Search Algorithm (GSA). The main idea is to integrate the ability of exploitation in PSO with the ability of exploration in GSA to synthesize both algorithms' strength. Some benchmark test functions are used to compare the hybrid algorithm with both the standard PSO and GSA algorithms in evolving best solution. The results show the hybrid algorithm possesses a better capability to escape from local optimums with faster convergence than the standard PSO and GSA.**

***Keywords-component; Gravitational Search Algorithm (GSA); Particle Swarm Optimization (PSO);Function optimization;***

## I. INTRODUCTION

In recent years, many heuristic evolutionary optimization algorithms have been developed. These include Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Deferential Evolution (DE), Ant Colony (AC), and Gravitational Search algorithm (GSA). The same goal for them is to find the best outcome (global optimum) among all possible inputs. In order to do this, a heuristic algorithm should be equipped with two major characteristics to ensure finding global optimum. These two main characteristics are exploration and exploitation.

Exploration is the ability of an algorithm to search whole parts of problem space whereas exploitation is the convergence ability to the best solution near a good solution. The ultimate goal of all heuristic optimization algorithms is to balance the ability of exploitation and exploration efficiently in order to find global optimum. According to [1], exploration and exploitation in evolutionary computing are not clear due to lake of a generally accepted perception. In other hand, with strengthening one ability, the other will weaken and vice versa.

Because of the above mentioned points, the existing heuristic optimization algorithms are capable of solving finite set of problems. It has been proved that there is no algorithm, which can perform general enough to solve all optimization problems [2]. Merging the optimization algorithms is a way to balance the overall exploration and exploitation ability. PSO is one of the most widely used evolutionary algorithms in hybrid methods due to its simplicity, convergence speed, and ability of searching global optimum.

There are some studies in the literature which have been done to synthesize PSO with other algorithms such as hybrid PSOGA [3,4], PSODE [5], and PSOACO [6]. These hybrid algorithms are aimed at reducing the probability of trapping in local optimum. Recently a novel heuristic optimization method is proposed called GSA [7]. In this study, we present a new hybrid model combining PSO and GSA algorithms named PSOGSA. We use 23 benchmark functions to compare the performance of hybrid algorithm with both standard PSO and standard GSA.

## II. THE STANDARD PSO AND STANDARD GSA

In this section, we provide a brief description of standard PSO and standard GSA.

### *A. Standard Particle Swarm Optimization*

PSO is an evolutionary computation technique which is proposed by Kennedy and Eberhart [8,9]. The PSO was inspired from social behavior of bird flocking. It uses a number of particles (candidate solutions) which fly around in the search space to find best solution. Meanwhile, they all look at the best particle (best solution) in their paths. In other words, particles consider their own best solutions as well as the best solution has found so far.

Each particle in PSO should consider the current position, the current velocity, the distance to *pbest*, and the distance to *gbest* to modify its position. PSO was mathematically modeled as follow:

$$v_i^{t+1} = wv_i^t + c_1 \times rand \times (pbest_i - x_i^t) + c_2 \times rand \times (gbest - x_i^t) \quad (1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

Where $v_i^t$ is the velocity of particle *i* at iteration *t*, *w* is a weighting function, $c_j$ is a weighting factor, rand is a random number between 0 and 1, $x_i^t$ is the current position of particle $i$ at iteration *t*, $pbest_i$ is the *pbest* of agent $i$ at iteration *t*, and *gbest* is the best solution so far.

The first part of (1), $wv_i^t$, provides exploration ability for PSO. The second and third parts, $c_1 \times rand \times (pbest_i - x_i^t)$ and $c_2 \times rand \times (gbest - x_i^t)$, represent private thinking and collaboration of particles respectively. The PSO starts with randomly placing the particles in a problem space. In each iteration, the velocities of particles are calculated using (1). After defining the velocities, the position of masses can be calculated as (2). The process of changing particles' position will continue until meeting an end criterion.

### *B. Standard Gravitational Search Algorithm*

GSA is a novel heuristic optimization method which has been proposed by E. Rashedi et al in 2009 [7]. The basic physical theory which GSA is inspired from is the Newton's theory that states: Every particle in the universe attracts every other particle with a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between them [10].

GSA can be considered as a collection of agents (candidate solutions) whose have masses proportional to their value of fitness function. During generations, all masses attract each other by the gravity forces between them. A heavier mass has the bigger attraction force. Therefore, the heavier masses which are probably close to the global optimum attract the other masses proportional to their distances.

The GSA was mathematically modeled as follow. Suppose a system with N agents. The algorithm starts with randomly placing all agents in search space. During all epochs, the gravitational forces from agent *j* on agent *i* at a specific time *t* is defined as follow [7]:

$$F_{ij}^{d}(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}\left(x_j^d(t) - x_i^d(t)\right), \quad (3)$$

Where $M_{aj}$ is the active gravitational mass related to agent *j*, $M_{pi}$ is the passive gravitational mass related to agent *i*, G(t) is gravitational constant at time *t*, $\varepsilon$ is a small constant, and $R_{ij}$ (t) is the Euclidian distance between two agents *i* and *j*.

The *G(t)* is calculated as (4):

$$G(t) = G_0 \times \exp\left(-\alpha \times iter/maxiter\right) \quad (4)$$

Where α and $G_0$ are descending coefficient and initial value respectively, *iter* is the current iteration, and *maxiter* is maximum number of iterations.

In a problem space with the dimension *d*, the total force that acts on agent *i* is calculated as the following equation:

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j F_{ij}^d(t), \quad (5)$$

Where $rand_j$ is a random number in the interval [0,1]. According to the law of motion, the acceleration of an agent is proportional to the result force and inverse of its mass, so the acceleration of all agents should be calculated as follow:

$$ac_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}, \quad (6)$$

Where *t* is a specific time and $M_i$ is the mass of object *i*.

The velocity and position of agents are calculated as follow:

$$vel_i^d(t+1) = rand_i \times vel_i^d(t) + ac_i^d(t), \quad (7)$$

$$x_i^d(t+1) = x_i^d(t) + vel_i^d(t+1), \quad (8)$$

Where $rand_i$ is a random number in the interval [0,1].

In GSA, at first all masses are initialized with random values. Each mass is a candidate solution. After initialization, velocities for all masses are defined using (7). Meanwhile the gravitational constant, total forces, and accelerations are calculated as (4), (5), and (6) respectively. The positions of masses are calculated using (8). Finally, GSA will be stopped by meeting an end criterion.

## III. The hybrid PSOGSA algorithm

Talbi in [11] has presented several hybridization methods for heuristic algorithms. According to [11], two algorithms can be hybridized in high-level or low-level with relay or co-evolutionary method as homogeneous or heterogeneous. In this paper, we hybridize PSO with GSA using low-level co-evolutionary heterogeneous hybrid. The hybrid is low-level because we combine the functionality of both algorithms. It is co-evolutionary because we do not use both algorithm one after another. In other words, they run in parallel. It is heterogeneous because there are two different algorithms that are involved to produce final results.

The basic idea of PSOGSA is to combine the ability of social thinking (*gbest*) in PSO with the local search capability of GSA. In order to combine these algorithms, (9) is proposed as follow:

$$V_i(t+1) = w \times V_i(t) + c_1' \times rand \times ac_i(t) + c_2' \times rand \times (gbest - X_i(t)) \quad (9)$$

Where $V_i(t)$ is the velocity of agent *i* at iteration *t*, $c_j'$ is a weighting factor, *w* is a weighting function, rand is a random number between 0 and 1, $ac_i(t)$ is the acceleration of agent *i* at iteration *t*, and g*best* is the best solution so far.

In each iteration, the positions of particles are updated as follow:

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (10)$$

In PSOGSA, at first, all agents are randomly initialized. Each agent is considered as a candidate solution. After initialization, Gravitational force, gravitational constant, and resultant forces among agents are calculated using (3), (4), and (5) respectively. After that, the accelerations of particles are defined as (6). In each iteration, the best solution so far should be updated. After calculating the accelerations and with updating the best solution so far, the velocities of all agents can be calculated using (9). Finally, the positions of agents are defined as (10). The process of updating velocities and positions will be stopped by meeting an end criterion. The steps of PSOGSA are represented in fig.1.

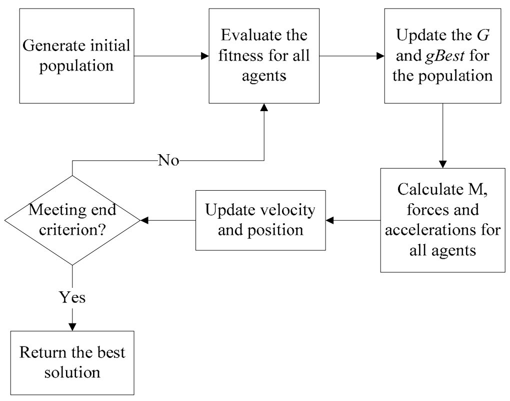
Generate initial population
Evaluate the fitness for all agents
Update the G and gBest for the population
Calculate M, forces and accelerations for all agents
Update velocity and position
Meeting end criterion?
No
Yes
Return the best solution

Figure 1. Steps of PSOGSA

To see how PSOGSA is efficient some remarks are noted as follow. In PSOGSA, the quality of solutions (fitness) is considered in the updating procedure. The agents near good solutions try to attract the other agents which are exploring the search space. When all agents are near a good solution, they move very slowly. In this case, the gBest help them to exploit the global best. PSOGSA use a memory (gBest) to save the best solution has found so far, so it is accessible anytime. Each agent can observe the best solution so far and tend toward it. With adjusting $c'_1$ and $c'_2$, the abilities of global search and local search can be balanced.

## IV. EXPERIMENTAL RESULTS AND DISCUSSION

To evaluate the performance of PSOGSA, it is applied to 23 standard benchmark functions [12]. Table I lists down these benchmark functions, the ranges of their search space, and their dimensions.

TABLE I. BENCHMARK FUNCTIONS

| Function | Dim[a] | Range |
|---|---|---|
| $F_1(x)=\sum_{i=1}^{n} x_i^2$ | 30 | [-100,100] |
| $F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$ | 30 | [-10,10] |
| $F_3(x)=\sum_{i=1}^{n}\left(\sum_{j-1}^{i} x_j\right)^2$ | 30 | [-100,100] |
| $F_4(x)=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$ | 30 | [-100,100] |
| $F_5(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right]$ | 30 | [-30,30] |
| $F_6(x)=\sum_{i=1}^{n}([x_i+0.5])^2$ | 30 | [-100,100] |
| $F_7(x)=\sum_{i=1}^{n} ix_i^4+random[0,1)$ | 30 | [-1.28,1.28] |
| $F_8(x)=\sum_{i=1}^{n}-x_i sin\left(\sqrt{\lvert x_i\rvert}\right)$ | 30 | [-500,500] |
| $F_9(x)=\sum_{i=1}^{n}[x_i^2-10cos(2\pi x_i)+10]$ | 30 | [-5.12,5.12] |
| $F_{10}(x)=-20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right)+20+e$ | 30 | [-32,32] |
| $F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right)+1$ | 30 | [-600,600] |
| $F_{12}(x)=\frac{\pi}{n}\{10sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n} u(x_i,10,100,4)$ | 30 | [-50,50] |
| $F_{13}(x)=0.1\{sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+sin^2(3\pi x_i+1)]+(x_n-1)^2[1+sin^2(2\pi x_n)]\}+\sum_{i=1}^{n} u(x_i,5,100,4)$ | 30 | [-50,50] |
| $F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$ | 2 | [-65.536, 65.536] |
| $F_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$ | 4 | [-5,5] |
| $F_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$ | 2 | [-5,5] |
| $F_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)cosx_1+10$ | 2 | [-5,5] |
| $F_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$ | 2 | [-2,2] |
| $F_{19}(x)=-\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j-p_{ij}\right)^2\right)$ | 3 | [1,3] |
| $F_{20}(x)=-\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j-p_{ij}\right)^2\right)$ | 6 | [0,1] |
| $F_{21}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$ | 4 | [0,10] |
| $F_{22}(x)=-\sum_{i=1}^{7}[(X-a_i)(X-a_i)^T+c_i]^{-1}$ | 4 | [0,10] |
| $F_{23}(x)=-\sum_{i=1}^{10}[(X-a_i)(X-a_i)^T+c_i]^{-1}$ | 4 | [0,10] |

a. Indicates the dimension of the function (the number of variables)

In this paper, our objective is minimization. The minimum values for functions appear in table I are 0 for $F_1$ to $F_{12}$ except $F_8$. The minimum values of $F_8$, and $F_{13}$ to $F_{23}$ are -12569.5, -1.15044, 0.998, 0.0003075, -1.0316, 0.398, 3, -3.86, 3.32, -10.2, -10.4, and -10.5, respectively.

PSO, GSA, and PSOGSA have several parameters which should be initialized. For PSO we use these settings: swarm size=30, $c_1$=2, $c_2$=2, $w$ is decreased linearly from 0.9 to 0.2, maximum iteration=1000, and stopping criteria=maximum iteration. For GSA and PSOGSA we use these settings: population size=30, $c'_1$=0.5, $c'_2$=1.5, $w$= random number in [0,1], $G_0$=1, $\alpha$=20 ,maximum iteration=1000, and stopping criteria=maximum iteration.

TABLE II. MINIMIZATION RESULTS OF BENCHMARK FUNCTIONS

| F | Std. PSO | | Std. GSA | | PSOGSA | |
|---|---|---|---|---|---|---|
| | *Ave* [a] | *Best* [b] | *Ave* | *Best* | *Ave* | *Best* |
| F1 | 2.83e-04 | 8.75e-06 | 1.19e-16 | 7.92e-17 | **6.66e-19** | **4.91e-19** |
| F2 | 5.50e-03 | 7.07e-06 | 4.77e-08 | 4.17e-08 | **3.79e-09** | **3.18e-09** |
| F3 | 5.19e+3 | 1.91e+3 | 734.566 | 297.666 | **409.936** | **43.2038** |
| F4 | 4.38e-07 | 2.41e-08 | 1.47e-02 | 9.72e-09 | **3.37e-10** | **2.96e-10** |
| F5 | 201.665 | **15.5933** | **35.0076** | 26.2566 | 56.2952 | 22.4221 |
| F6 | 4.96e+0 | 4.51e+0 | 1.67e-16 | 8.17e-17 | **7.40e-19** | **5.76e-19** |
| F7 | 2.60e-01 | 1.05e-01 | 4.58e-01 | 8.07e-02 | **5.09e-02** | **2.77e-02** |
| F8 | -5909.47 | 7802.34 | -2437.52 | -3127.8 | **-12213.7** | **-12569** |
| F9 | 72.9581 | 55.7182 | 31.1185 | 24.1444 | **22.6777** | **19.1371** |
| F10 | 4.85e-10 | **2.48e-12** | 7.66e-09 | 5.57e-09 | **6.68e-12** | 5.97e-12 |
| F11 | 5.43e-03 | 9.96e-07 | 6.98e+0 | 3.96e+0 | **1.48e-03** | **1.11e-16** |
| F12 | 2.29e00 | **1.06e-01** | **1.95e-01** | **1.06e-01** | 23.4e+0 | 6.43e+0 |
| F13 | **8.97e-02** | **1.10e-02** | 3.30e-03 | **1.1e-02** | 7.78e-19 | 7.87e-19 |
| F14 | **0.998** | **0.998** | 3.14 | 1.0003 | 1.49 | **0.998** |
| F15 | 1.04e-03 | 9.77e-04 | 5.33e-03 | 2.50e-03 | **8.56e-04** | **3.07e-04** |
| F16 | **-1.0316** | **-1.0316** | **-1.0316** | **-1.0316** | **-1.0316** | **-1.0316** |
| F17 | **39.79** | **39.79** | **39.79** | **39.79** | **39.79** | **39.79** |
| F18 | **3** | **3** | **3** | **3** | **3** | **3** |
| F19 | **-3.8628** | **-3.8628** | -3.8625 | -3.86019 | **-3.8628** | **-3.8628** |
| F20 | -1.60048 | **-2.9587** | **-1.65077** | -2.2641 | -8.92e-1 | -2.6375 |
| F21 | -6.5752 | **-10.1532** | -3.66413 | -5.05520 | **-7.25959** | **-10.1532** |
| F22 | -8.05697 | **-10.4029** | **-10.4028** | **-10.4029** | -7.53978 | **-10.4029** |
| F23 | -7.33602 | **-10.5364** | **-10.5363** | **-10.5364** | -7.52233 | **-10.5364** |

a. Indicates average of best function values in 30 runs
b. Indicates the best result in 30 runs

The experimental results are presented in table II. The results are averaged over 30 runs and the best results are indicated in bold type.

Statistically speaking, for the best of 30 runs on 23 test functions, PSOGSA is the best on 18 functions, PSO is the best on 13 functions, and GSA is the best on 7 functions.

For the average best of 30 runs on 23 test functions, the PSOGSA performs the best on 16 functions, PSO is the best on 6 functions, and GSA is the best on 8 functions.

Therefore, the number of function which the PSOGSA performs better is close to twice as many as PSO and GSA.

For the average best of 30 runs, PSOGSA reaches global minima in all of the benchmark functions except $F_5$, $F_{12}$, $F_{13}$, $F_{14}$, $F_{20}$, $F_{22}$, and $F_{23}$. Function $F_5$ is a noisy problem, and all algorithms have similar convergence patterns on it. For the functions $F_{14}$, $F_{22}$, and $F_{23}$, PSOGSA can find the global minima in 30 runs as PSO and GSA. For the functions $F_{14}$, $F_{20}$, $F_{22}$, and $F_{23}$, they all have narrow domains, so it can be concluded that PSOGSA performs better on functions with wide domains. For the high-dimensional functions ($F_1$ to $F_{13}$), it is hard for PSOGSA to find global minima only on $F_{12}$ and $F_{13}$, so it can be also concluded that PSOGSA has good performance on high-dimensional functions.

The following figures of selected functions also show that new PSOGSA performs better than standard PSO and GSA in terms of convergence rate.

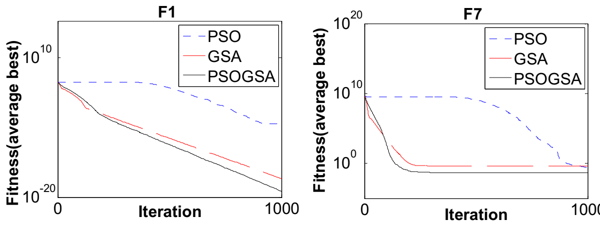

Figure 2. Average best of benchmark functions F1 and F7

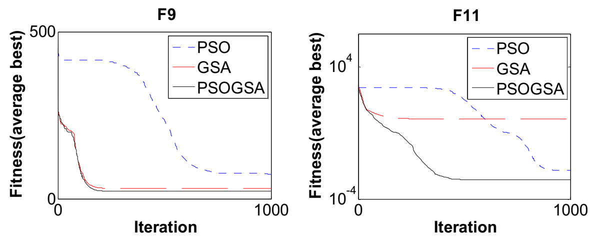

Figure 3. Average best of benchmark functions F9 and F11

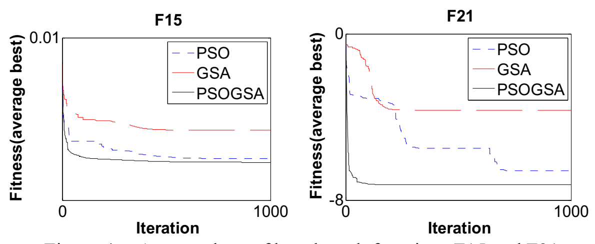

Figure 4. Average best of benchmark functions F15 and F21

## V. CONCLUSION

In this research, a new hybrid algorithm is introduced utilizing strengths of PSO and GSA. The main idea is to integrate the abilities of PSO in exploitation and GSA in exploration. Twenty-three benchmark functions are used to validate the performance of the PSOGSA compared to standard PSO and GSA. The results show that PSOGSA outperforms both in most function minimization. The results are also proved that the convergence speed of PSOGSA is faster that PSO and GSA.

## ACKNOWLEDGMENT

This work is supported by a research grant from Ministry of Science, Technology and Innovation (MOSTI), Malaysia through E-Science Fund Scheme project number 01-01-06-SF0530, VOTE 79332. The authors gratefully acknowledge many helpful comments by reviewers and members of SCRG in improving the publication.

## REFERENCES

[1] A.E. Eiben and C.A. Schippers, "On evolutionary exploration and exploitation," Fundamenta Informaticate, vol. 35, no. 1-4, pp. 35-50, 1998.

[2] DH. Wolpert and WG. Macread, "No free lunch theorems for optimization," IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 67-82, 1997.

[3] X. Lai and M. Zhang, "An efficient ensemble of GA and PSO for real function optimization," in 2nd IEEE International Conference on Computer Science and Information Technology, 2009, pp. 651-655.

[4] A. A. A. Esmin, G. Lambert-Torres, and G. B. Alvarenga, "Hybrid Evolutionary Algorithm Based on PSO and GA mutation," in proceeding of the Sixth International Conference on Hybrid Intelligent Systems (HIS 06), 2007, p. 57.

[5] L. Li, B. Xue, B. Niu, L. Tan, and J. Wang, "A Novel PSO-DE-Based Hybrid Algorithm for Global Optimization," in Lecture Notes in Computer Science.: Springer Berlin / Heidelberg, 2008, pp. 785-793.

[6] N. Holden and AA Freitas, "A Hybrid PSO/ACO Algorithm for Discoverin Classification Rules in Data Mining," Journal of Artificial Evolution and Applications (JAEA), 2008.

[7] E. Rashedi, S. Nezamabadi, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," Information Sciences, vol. 179, no. 13, pp. 2232-2248, 2009.

[8] J. Kennedy and RC. Eberhart, "Particle swarm optimization," in Proceedings of IEEE international conference on neural networks, vol. 4, 1995, pp. 1942–1948.

[9] Y. Shi and R.C. Eberhart, "A modified Particle SwarmOptimiser," in IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, 1998.

[10] Isaac Newton, In experimental philosophy particular propositions are inferred from the phenomena and afterwards rendered general by induction, 3rd ed.: Andrew Motte's English translation published, 1729, vol. 2.

[11] E. G Talbi, "A Taxonomy of Hybrid Metaheuristic," Journal of Heuristics, vol. 8, no. 5, pp. 541-546, 2002.

[12] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," IEEE Transactions on Evolutionary Computation, vol. 3, pp. 82-102, 1999.