A Novel Hybrid Gravitational Search Particle Swarm Optimization Algorithm

*Talha Ali Khan, Sai Ho Ling, and A.S. Mohan
School of Biomedical Engineering, University of Technology Sydney, New South Wales, Ultimo, 2007, Australia
*Talha.khan@uts.edu.au

Abstract— **Particle Swarm Optimization (PSO) algorithm is a member of the swarm computational family and widely used for solving nonlinear optimization problems. But, it tends to suffer from premature stagnation, trapped in the local minimum and loses exploration capability as the iteration progresses. On the contrary, Gravitational Search Algorithm (GSA) is proficient for searching global optimum, however, its drawback is its slow searching speed in the final phase. To overcome these problems in this paper a novel Hybrid Gravitational Search Particle Swarm Optimization Algorithm (HGSPSO) is presented. The key concept behind the proposed method is to merge the local search ability of GSA with the capability for social thinking (gbest) of PSO. To examine the effectiveness of these methods in solving the abovementioned issues of slow convergence rate and trapping in local minima five standard and some modern CEC benchmark functions are used to ensure the efficacy of the presented method. Additionally, a DNA sequence problem is also solved to confirm the proficiency of the proposed method. Different parameters such as Hairpin, Continuity, H-measure, and Similarity are employed as objective functions. A hierarchal approach was used to solve this multi-objective problem where a single objective function is first obtained through a weighted sum method and the results were then empirically validated. The proposed algorithm has demonstrated an extraordinary performance per solution stability and convergence.**

Keywords—PSO; GSA; Hybrid; DNA Computation.

I. INTRODUCTION

During the last few years, many new and modified versions of the existing algorithms have been the topic of interest in the evolutionary computational research community. These algorithms include Particle Swarm Optimization Algorithm, Genetic Algorithm, Ant Colony, Gravitational Search Algorithm to name a few. The primary focus of all these algorithms is to trace the best solution and to evade trapping in the local minima. The algorithm that is used for obtaining the optimized solution must contain the main two characteristics these are exploration during global searching and exploitation during the local search.

To obtain a good heuristic optimization method there must be a steadiness between exploitation and exploration. Exploitation is the convergence proficiency of the algorithm to the best solution whereas exploration is defined as the competency to search the entire area of the problem space. The primary objective of the optimization algorithms is to maintain a trade-off between exploitation and exploration effectively to obtain the global optimum. However, it is challenging to keep a balance between exploration and exploitation as one ability often overlaps the others [1]. Therefore, due to this problem, the current optimization algorithms are only proficient in solving a limited number of problems. Until now, no optimizing algorithm is capable of solving all the problems. The amalgamation of the different optimization algorithms is one of the other approaches to maintain a steadiness between exploitation and exploration capability. PSO is a nonlinear smart computational technique that is undeterred by the problem size, and effectively utilized to obtain an optimal solution compared to other conventional techniques [2]. Therefore, it can be used proficiently with numerous optimization problems. Previously many other optimizing algorithms are combined with PSO to minimize the trapping probability in the local minimum. Lately, a unique optimization technique based on the force of gravity is been presented known as GSA. Therefore, in this paper, a combination of both these techniques has been used.

A. Standard Particle Swarm Optimization Algorithm (SPSO):

Eberhart and Kennedy in 1995 [3], proposed the PSO algorithm that depends on the concepts of the social behavior of the birds. In the same way as the flock of birds, the algorithm comprises the number of particles to form a swarm. Within the searching area, each agent is finding the best solution.

At the initial stage, a swarm is formed by allocating random velocity and positions to all the particles. The assessment of each agent (particle) fitness is completed by a set test benchmark function. The velocity and the position after every iteration is calculated by using (1) and (2). Accordingly, if the position found out is improved than the last best position is stored in the memory. V_{max} is set to limit the redundant mobility of the agents outside the search zone. If the velocity goes above v_{max} it is set to zero. Every particle travel in the search space for calculating the best solution. The position of each particle is calculated as

$$x_i(t+1) = x_i(t) + v_i(t) \quad (1)$$

The information of every particle is based upon its knowledge and the surrounding particle's experience. These fundamentals have identical significance and might

be altered based on the choice of the particle so the velocity equation will be

$$v_i = \chi.\{w.v_i + C_1R_1(P_i - x_i) + C_2R_2(P_g - x_i)\} \quad (2)$$

where

$P = [P_1, P_2, P_3, P_D]$

$P_g = [Pg_1, Pg_2, Pg_3, Pg_D]$

i=1, 2,… D.

P_g is the global best position, the v_i = {v_1, v_2... v_n} is the velocity of the particles, *R* is the random number [0-1], *D* is the dimension of the search space $D \in \{1, 2, 3….D\}$, P_i is the local best, and x_i is the current position. Each particle is assessed by a given fitness function. The primary purpose of the PSO is to decrease the cost values of the particles iteratively for the given benchmark function.

B. Standard Gravitational Search Algorithm (SGSA):

Rashedi et.al proposed a novel method known as Gravitation search optimization (GSA) in 2009 [4]. The concept behind this algorithm was dependent on Newton's law of motion and gravity. Every agent has its mass in GSA, and the heavier mass agent gives a better force of attraction. Therefore, all the agents travel near to the heaviest agent as the iteration progresses.

The position of the agents is initialized as follows:

$$X_i = \left(X_i^1, X_i^2, \dots\dots\dots\dots\dots X_i^d, X_i^D\right) for\ i = (1,2,..D) \quad (3)$$

In the D^{th} dimension, x_d denotes the position of an i^{th} object, and *D* denotes the dimension of the search area.

At time "*t*" the force of attraction by mass "*i*" from mass "*j"* is defined as:

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) * M_{aj}(t)}{R_{ij}(t) + \epsilon}\ (X_j^d(t) - X_i^d(t)) \quad (4)$$

$$G(t) = G(t_0) * \left(\frac{t_0}{t}\right)^{\beta}\ , \beta < 1 \quad (5)$$

where M_{pi} is the passive gravitational mass related to object *i*, $R_{ij}(t)$ is the distance between two agents *i* and *j,* ε is a small constant, $G(t)$ is a gravitational constant at time *t,* and M_{aj} is an active gravitational mass of object *j*.

In dimension "D" the overall force experienced on agent *i* is a random weighted sum of all the other parts of the forces exerted on the other agents, (4) can be modified by the following equation.

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j\ F_{ij}^d(t) \quad (6)$$

According to the law of motion, the acceleration of the agent can be defined as:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \quad (7)$$

Where M_{ii} denotes the inertial mass.

The velocity of the masses is dependent on their current velocity and acceleration. The velocity and position of the agent is updated by (8) and (9)

$$X_i^d(t+1) = X_i^d + v_i^d(t+1) \quad (8)$$

$$V_i^d(t+1) = rand \times v_i^d(t) + a_i^d(t) \quad (9)$$

where "*t"* is the present iteration. Let's suppose that the gravitational and inertial masses are equal then masses will be calculated as follows:

$$M_{ai} = M_{ii} = M_{pi} = M_i \qquad i = 1,2,3, \dots\dots\dots N \quad (10)$$

$$m_i(t) = \frac{fit_i(t) - X^w(t)}{X^b(t) - X^w(t)} \quad (11)$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \quad (12)$$

Where $fit_i(t)$ is the optimal value of the agent and $X^w(t)$ is the worst and $X^b(t)$ *is* the best value for this function in case of the minimization problem it can be calculated as:

$$X^b(t) = min_{j\,\epsilon\,\{1,\dots N\}}\ fit_j(t) \quad (13)$$

$$X^w(t) = max_{j\epsilon\,\{1,\dots N\}}\ fit_j(t) \quad (14)$$

To maintain a trade-off between exploitation and exploration of the algorithm over the iteration the number of the agents is reducing so that the agents with heavier masses acting force to one another are evaluated. "K_{best}" are the best agents having heavier masses. Therefore, the value of the K_{best} is reducing slowly until a single agent is exerting force to the remaining agents. Consequently, (4) is modified as:

$$F_i^d(t) = \sum_{j\,K_{best}, j\neq i}\ rand_j\ F_{ij}^d(t) \quad (15)$$

C. Amendments and Contributions in this work:

Although the proficiency has been enhanced by these several versions of PSO and GSA most of these variants do not have a strong learning approach for the particles that perform well during the searching phase. On the other hand, those particles that have not improved their fitness over the iterations and may have suffered from stagnation, and no technique is used to solve this condition as well. These concerns may limit the ability of the PSO and GSA algorithms for solving highly complex global optimization problems.

Since both the algorithms have some points in common and a few differences, so it is worth mentioning it before proposing the hybrid version. These are highlighted as under:

- Both the algorithm come under the population-based category.
- PSO depends on the social behaviour of the birds and flocks of fishes whereas the GSA depends on the laws of physics.

- The optimization is achieved in both the algorithms by the agents' mobility in the search area, though the mobility scheme is dissimilar.
- The total force of attraction acted by all other agents assesses the direction of the agents in GSA, whereas, the direction of the particles is obtained by global best and local best positions in PSO.
- In GSA, the updating process is carried out by considering the force of attraction experienced by agents that is proportional to the fitness value, therefore, the agents look into the search area around them under the effect of force. On the contrary, in PSO, updating evaluation is achieved lacking the quality of the solutions, and considering the fitness values as insignificant in the updating process.
- GSA is a memoryless algorithm and only the present position is important in the updating process, but for updating the velocity in PSO, it utilizes a kind of memory.
- The distance between the particles is not important in PSO for updating. On the other hand, the force is inversely proportional to the distance between the agents in GSA.

In this paper, the following modifications and contributions have been presented:

- A new method is implemented for the velocity clamping by adding a new velocity term in the velocity update equation so that the particles can move faster towards the optimal solution. Since the particle velocity is a presumptive parameter, so, it forms an uncurbed path permitting the particles to create wider cycles in the search area. To circumvent these oscillations, the limits for the velocity are implemented in terms of upper and lower bounds in the HGSPSO. One other primary factor for amending the velocity update equation is that during exploration, the particles trapped in the local minima and cannot leave which resulted in a premature convergence. The concept of increasing the velocity of the particles helps to reach the optimal position swiftly and ameliorate the convergence.
- Adaptive time-varying acceleration constants presented for changing the constants as the iteration progresses.
- Inertial weight is also introduced in this paper that makes a balance between the exploration and exploitation of the HGSPSO.
- The proposed method is applied to real-world DNA problem to demonstrate the performance of the HGSPSO.

So, in this paper, a fusion of the GSA and PSO is presented that uses the exploitation capability of the PSO and exploration proficiency of the GSA. The efficiency of the HGSPSO is then compared with the standard versions of both GSA and PSO by using five standard test benchmark functions, CEC benchmark functions, and solving a DNA problem. The layout of the remaining paper is as follows. Section II contains the hybrid gravitational search particle swarm optimization algorithm discussion. Section III shows the application of the proposed method on the benchmark functions, Section IV contains the fundamental concepts of the DNA; and the comparison of the presented technique is made with the contemporary algorithms for DNA computation. The paper is culminated by presenting the key conclusions in section V.

II. HYBRID GRAVITATIONAL SEARCH PARTICLE SWARM OPTIMIZATION ALGORITHM (HGSPSO)

Many hybrid algorithms are available in the literature that use the properties of two or more algorithms to improve the capability of the hybrid version. Different approaches are used to combine the two algorithms that are at the low level and high level in a homogenous and heterogeneous way. In this paper, GSA is combined with PSO, and the properties of both the algorithms are utilized i.e. both the algorithms were executed at the same time and the hybrid algorithm i.e. HGSPSO is utilized for obtaining the results. The key rationale for the proposed fusion is to benefit from the exploitation capability of the PSO and exploration proficiency of the GSA.

In HGSPSO, the parameters of both these algorithms are used. The inertial weight and acceleration constants are the two important parameters in PSO. Therefore, in HGSPSO inertial weight "*w*" are modified in such a way that either it is not used as a linear reducing or not it is set as a constant value, however, it is used as a function of the fitness function values of the global and the local best. Similarly, the acceleration constants are used in this paper are also adaptively changes as the iteration progresses. These two parameters are defined as follows:

$$w_i = 1 - \left(\frac{P_g}{P_i}\right) \tag{16}$$

$$C_1 = (C_3 - C_4) * (1 - \frac{t}{T}) + C_4 \tag{17}$$

$$C_2 = (C_5 - C_6) * (1 - \frac{t}{T}) + C_6 \tag{18}$$

where C_3 C_4, C_5, and C_6 are constants values, "*T*" is the maximum iteration, and "*t*" is the current iteration. The values for C_1 are reducing adaptively and C_2 is increasing over the iteration, as a result, the masses move closer to the best solution as the proposed method moves in the exploitation stage. The reason for adjusting the acceleration coefficients adaptively as there is no specific boundary in the evolutionary computation for the transiting between these two stages. Moreover, this adaptive method helps the exploration in the beginning and exploitation in the later stage. The best results of the HGSPSO are obtained when changing the C_1 from 2.5 to 0.5, C_2 from 0.5 to 2.5, P_g is the global best, and P_i is the local best.

In HGSPSO, every agent is considered as a potential solution. The gravitational constant, resultant forces and gravitational force are measured among other agents through (19), (20), and (21). (22) calculates the acceleration of the agents.

At time "*t*" the force of attraction by agent (mass) "*i*" from an agent (mass) "*j*" is defined as:

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) * M_{aj}(t)}{R_{ij}(t) + \varepsilon} \left(X_j^d(t) - X_i^d(t)\right) \tag{19}$$

where M_{pi} is the passive gravitational mass associated to object i, ε is a small constant, $R_{ij}(t)$ is the distance between two agents i and j, M_{aj} is an active gravitational mass of agent j, and $G(t)$ is a gravitational constant at time t.

To signify the importance of the exploration in the initial phase of the algorithm and exploitation in the later phase, G has been used adaptively so its value rises as the iteration increases. The main objective for designing the G adaptively is to motivate the agents to move with bigger steps in the early stage of the algorithm, however, agents are bound to travel gradually at the end of the iterations. $G(t)$ is modified as follows:

$$G(t) = G_o \times e^{-\gamma \times (\frac{T-t}{T})} \quad (20)$$

where G_0 is the initial gravitational constant and γ is the coefficient of decrease.

Assume that in dimension "D" the overall force experienced on agent i is a random weighted sum of all the other parts of the forces exerted on the other agents, (19) can be modified by the following equation.

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j F_{ij}^d(t) \quad (21)$$

Similarly, acceleration can be calculated as:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \quad (22)$$

The gravitational masses, inertial mass for the minimization problem can be calculated in the same way as defined from (10)-(14).

As discussed earlier, that the GSA is a memoryless algorithm therefore the best solution might not be saved as the best mass is attracted away by other less fitted masses. To overcome This limitation is mitigated through proposing a novel strategy in the velocity update procedure. The new equation for the velocity update is defined as:

$$V_i(t+1) = w \times v_i(t) + C_1 \times a_i(t) + \left(\frac{C_1}{C_2}\right) \times \left(p_g - P_i\right) \quad (23)$$

Where C_1 and C_2 are the acceleration constants, w is the inertial weight, a is the acceleration, p_g is the global best (g_{best}), P_i is the local best position v_i is the velocity of the agent. The third term in (23) is somewhat analogous to the social term of the PSO velocity equation. A higher value of C_1 biases towards GSA behavior, whereas a higher value of C_2 encourages the social factor of PSO in the execution of the search procedure. The adaptive technique permits GSA to examine the search area more effectively and a PSO alike exploitation of the best solution.

Finally, the position of agents is updated as follow:

$$X_i^d(t+1) = w * X_i^d + v_i^d(t+1) \quad (24)$$

In the presented algorithm, all agents are initialized randomly. The gravitational force, gravitational constant and resulting forces between them are obtained using (19), (20) and (21), correspondingly. Later, the acceleration of particles is measured using (22). At each iteration, the best solution attained up to now should be updated. Then, the velocities of all agents are computed using (23), and lastly, (24) update the positions of agents. The procedure ends after reaching the desired stopping criteria.

A few advantages and remarks on the proposed method are highlighted as under:

- The presented algorithm used memory for storing the best solution achieved, in the updating process the quality of the agent's solutions is also considered. The particles that are closer to the potential solutions attempted to allure the other particles that are exploring the search space. In the process of exploring the search area, the particles that are, closer to the potential solution started to move slowly toward the optimal point. The effectiveness of the global best searching topology criteria helps the particles to explore the global best. Since HGSPSO utilizes a memory-based global best topology to save the potential solution available so the best solution will not be missing and will be easily available at any point in time.

- Each particle will explore the best solution and travel in the direction of it, so masses are providing with a kind of social intelligence.

- The computational cost of this algorithm is very low.

- The influence of P_g is noticeable in the exploitation stage by using adaptive C_1 and C_2.

- The impact of P_g on agents is free of their masses and it will be considered as an exterior force not depending on gravitational rules. This efficiently stops particles from get-together and having very slow movement.

Due to these modifications in the presented algorithm, it delivers better results than the standard GSA and PSO.

III. HGSPSO FOR FUNCTIONS OPTIMIZATION

Five test benchmark functions [5] are used to confirm the efficiency of the proposed method.

A. Standard Benchmark Functions

The strength, efficacy, and ability tests of different optimization approaches are demonstrated by utilizing numerous standards and benchmark functions. Thus, the solution quality, convergence, and stability are measured. To assess the proposed algorithm proficiency few standard test benchmark functions are used.

i) Sphere:

It is a unimodal function with single minima. The main objective of using a sphere benchmark is to test the convergence rate of the algorithm.

$$f_1(x) = \sum_{i=1}^{D} x_i^2 \quad (25)$$

ii) Rastrigin:

It's a multi-modal function comprising many local minima.

$$f_2(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10] \quad (26)$$

iii) Rosenbrock:

It is a unimodal function with a single minimum.

$$f_3(x) = \sum_{i=1}^{D-1}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \quad (27)$$

iv) Griewank

It is a multi-modal function with many local minima as a result; it tends to convergence in the wrong direction.

$$f_4(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1 \quad (28)$$

v) Ackley:

Ackley is a multi-modal function with many local minima.

$$f_5(x) = -20exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e \quad (29)$$

1) Comparison with the state-of-the-art algorithms:

The comparison is made with the standard PSO, GSA, and the following techniques.

a) Improved Hybrid Gravitational Search Algorithm (IHGSA) [6]:

In IHGSA, a new parameter that is known as the learning factor is introduced. It helps to maintain a balance between exploitation and exploration of this method.

b) Mean Gbest Particle Swarm Optimization Gravitational Search Algorithm (MGBPSO-GSA) [7]:

MGBPSO-GSA is a hybrid version of the Mean Gbest Particle swarm optimization algorithm and the gravitational search algorithm.

c) Self-learning Particle Swarm Optimizer (SLPSO)[8]:

Li et.al introduced a new four different learning approaches where every particle adaptively adopts one according to the local fitness landscape.

d) Dynamic Neighborhood learning-based GSA(DNLGSA)[9]:

Aizhu et. Al presented a dynamic neighborhood learning (DNL) strategy to replace the K_{best} model to maintain a balance between exploration and exploitation in GSA.

e) Dynamic multi swarm particle swarm optimization and gravitational search algorithm (GSADMSPSO)[10]:

In this method, the main population of masses is divided into smaller sub-swarms and also stabilizing them by presenting a new neighborhood strategy.

f) Hybrid Gravitational Search Algorithm [11]:

A local search technique (LST) is combined with the optimization procedure of the GSA. Every agent in GSA has a probability (p), and LST with probability ($1-p$). Fuzzy logic is used to get the probability p.

2) Results and analysis.

Termination criteria are that the global optimum is found or the number of iterations is reached. Experiments are run in MATLAB on a PC with the 64-bit win10 professional operating system, 16 GB RAM, and 2.90GHz processor. The values of the parameters of the compared algorithms are taken from their respective literature as shown in table I. The simulation conditions used for the algorithm are:

- Dimension D=30; G_0 = 1
- No. of agents= 30

Table I Parameters of the compared algorithms

Algorithms	Parameters
IHGSA	Swarm size= 30, G_0 = 1, α = 20, learning factors S_1 and S_2 of IHGSA are in the range of [0.618, 1.236].
MGBPSO-GSA	G_o=1; Swarm Size=30; C_1=0.5; D=30, and C_2=1.5.
SLPSO	*Swarm Size= 30,* P=1, γ=0.01
DNLGSA	G_o=100, β=20, k=10, gm=5, $C_1 = 0.5 - \frac{0.5t^{0.166}}{T_{max}{}^{0.166}}$, $C_2 = \frac{1.5\,t^{0.166}}{T_{max}{}^{0.166}}$, D=30
GSADMSPSO	Swarm size=30, C_1 = 0.5, C_2 = 1.5, w is decreased linearly from 0.9 to 0.2, G_0 = 1, α = 20, R=5.
HGSA	G_o=100, α = 20, θ= (0.5) ^{D}, γ=0.2

a) Statistical Analysis:

HGSPSO results are equated with the SGSA, SPSO, different variants of the modified PSO, and improved GSA algorithms. Mean and standard deviation are used as the evaluating criteria as shown in table II.

i) T-Test:

T-Test value is one of the main testing criteria to compare the two algorithms. Mean "α", ξ value of the degree of freedom, and standard deviation "σ" values of the comparing methods are utilized to find out the t-test value. The negative values of the test show the inferior performance of the first approach than the second method or vice versa. The t-value can be calculated as:

$$t = \frac{\alpha_1 - \alpha_2}{\sqrt{\left(\frac{\sigma_1^2}{\xi+1}\right)+\left(\frac{\sigma_2^2}{\xi+1}\right)}} \quad (30)$$

Larger values of the t-test than 1.645 implies enhanced performance of the first algorithm as compared to the second. The t-values for the proposed method compare with the SGSA, SPSO, and other algorithms are summarized in Table II.

ii) Wilcoxon Signed-Rank Test:

The standard deviation and mean values of every method in the assumed function evaluations are used to assess the final solution's quality. Furthermore, the Wilcoxon signed-rank test at a 0.05 significance level (α) is used to analyze the dissimilarity between the two methods. Table III illustrates the results attained by the Wilcoxon signed-rank test that depends on the results of tables II. Where *win*, *tie*, and *lose* show that HGSPSO, successes on w functions, draws on t functions, and fails on l functions than the other methods

TABLE II. The standard deviation, t-test and mean values evaluation between HGSPSO and other approaches (All Results Are Averaged (Rank: 1—Best, 9—Worst)

Functions		HGSPSO c_1= [2.5,0.5] c_2= [0.5,2.5]	SPSO	SGSA	IHGSA	MGBPSO-GSA	SLPSO	DNLGSA	GSADMSPSO	HGSA
$f_1(x)$	Mean	**$6.256e^{-52}$**	$1.821e^{-8}$	$1.71e^{-06}$	$2.38e^{-19}$	$2.58e^{+03}$	$2.78e^{-50}$	$1.53e^{-27}$	$1.50e^{-26}$	0.0000
	Std. Dev	**$9.293e^{-53}$**	$2.664e^{-8}$	$7.2e^{-07}$	$5.58e^{-20}$	159.203	$8.09e^{-49}$	$5.11e^{-26}$	$5.73e^{-25}$	0.0000
	t-test	-	4.8335	16.793	30.1598	114.59	0.2375	0.2117	0.1851	0.000
	Rank	***1***	6	7	5	8	2	3	4	1
$f_2(x)$	Mean	**$1.095e^{-8}$**	20.392	$1.94e^{+2}$	$3.38e^{1}$	41.5262	0.000	0.000	8.19	0.000
	Std. Dev	**0.25302**	5.389	$1.08e^{+2}$	$1.05e^{1}$	10.4930	0.000	0.000	6.11	0.000
	t-test	-	26.72	12.70	22.75	27.98	0.000	0.000	9.4701	0.000
	Rank	***1***	2	6	5	4	1	1	3	1
$f_3(x)$	Mean	**$1.341e^{-08}$**	$3.3426e^{1}$	$1.70e^{+2}$	$2.28e^{+1}$	$9.164e^{6}$	2.06	$2.33e^{1}$	$2.38e^{1}$	23.7073
	Std. Dev	**$6.508e^{-09}$**	2.1794	2.12	1.29	$3.46e^{5}$	12.3	$3.24e^{1}$	$3.56e^{0}$	0.3860
	t-test	-	108.39	567.01	124.97	187.28	1.184	5.085	47.272	4.3415
	Rank	***1***	7	8	3	9	2	4	6	5
$f_4(x)$	Mean	**$3.884e^{-24}$**	$9.44e^{-2}$	$2.4589e^{-3}$	$1.55e^{-16}$	18.1456	0.0227	0.000	$7.40e^{-3}$	0.0000
	Std. Dev	**$7.128e^{-24}$**	$1.025e^{-1}$	$2.53e^{-3}$	$1.02e^{-16}$	1.1171	0.144	0.000	$2.31e^{-2}$	0.0000
	t-test		6.512	6.845	10.74	114.58	1.1146	0.000	2.265	0.000
	Rank	***1***	6	4	2	7	5	1	3	1
$f_5(x)$	Mean	**$2.339e^{-42}$**	$4.5989e^{-3}$	$1.54e^{-06}$	$6.58e^{-10}$	1.4094	$3.47e^{-14}$	$6.8e^{-14}$	$1.84e^{-14}$	$8.881e^{-16}$
	Std. Dev	**$9.426e^{-42}$**	$3.2724e^{-3}$	$6.78e^{-07}$	$5.22e^{-11}$	0.2447	$4.45e^{-14}$	$5.46e^{-14}$	$4.80e^{-13}$	$2.742e^{-30}$
	t-test	-	9.938	16.061	89.133	40.72	5.5138	8.8064	0.2710	$2.2902e^{15}$
	Rank	***1***	8	7	6	9	4	5	3	2
Overall Ranking (Average Ranking Number)		**1 (1.0)**	6 (5.8)	7 (6.4)	5 (4.2)	8 (7.4)	3 (2.8)	3 (2.8)	4 (3.8)	2 (2.0)

TABLE III results of Wilcoxon signed-rank test between HGSPSO and other methods on 30-*D* benchmark functions

HGSPSO v.s.	SPSO	SGSA	IHGSA	MGBPSO-GSA	SLPSO	DNLGSA	GSADMSPSO	HGSA
w (+)	5	5	5	5	4	3	5	2
t (=)	0	0	0	0	1	2	0	3
l (-)	0	0	0	0	0	0	0	0

b) Parameters analysis of HGSPSO:

To evaluate the performance of the proposed method testing is done on the different values of the design variables. The tuning of the parameters of the proposed method such as the acceleration constants are examined on the several ranges of C_1 and C_2. The values of the C_1 and C_2 are chosen in such a way that it will fulfill the HGSPSO convergence condition $C_1+C_2 \leq 4$ [12]. Another parameter that influences the performance of the HGSPSO is the inertial weight. Shi et al. [13]used different constant values for inertial weight and highlight that larger values of the inertial weight i.e. $w > 1.2$, tends the PSO to performs a weak exploration, on the other hand, smaller values of inertial weight, i.e. $w < 0.8$, make the PSO to traps in local optima. They concluded that the constant values of the inertial weight must be within the bounds of [0.8, 1.2]. Therefore, in this paper for making a comparison of the proposed adaptive inertial weight strategy and the constant inertial weight, the value of the inertial weight is set to $w=0.9$. A test is conducted on some unimodal and multimodal function by using different ranges of the C_1 and C_2, a constant inertial weight value, and adaptive inertial weight approach by using (16). Table IV shows the study of acceleration constants at different values for unimodal and multimodal functions and by applying a constant value of "w" and adaptive approach. The best results of the HGSPSO are obtained when changing the C_1 from 2.5 to 0.5 and C_2 from 0.5 to 2.5 and having adaptive inertial weight. So, these values are used for the rest of the work.

Table IV Analysis of acceleration constants variations and inertial weight at different values for D=30

HGSPSO	c_1= *[2.5,0.5]* c_2= *[0.5,2.5]*	c_1= *[2.0,0.0]* c_2= *[0.0,2.0]*	c_1= *[2.0,0.25]* c_2= *[0.25,2.0]*
$f_1(x)$ w= using (16)	Mean= $6.256e^{-52}$ SD= $9.293e^{-53}$	Mean= $1.804e^{-10}$ SD= $5.308e^{-10}$	Mean= $7.442e^{-28}$ SD= $2.175e^{-28}$
$f_1(x)$ w= *1.0*	Mean= $9.386e^{-22}$ SD= $3.458e^{-22}$	Mean= $3.418e^{-2}$ SD= $1.994e^{-2}$	Mean= $6.463e^{-14}$ SD= $5.335e^{-14}$
$f_3(x)$ w= using (16)	Mean=$1.341e^{-08}$ SD=$6.508e^{-09}$	Mean=$9.04e^{1}$ SD= $8.22e^{1}$	Mean=$4.422e^{-2}$ SD=$7.093e^{-2}$
$f_3(x)$ w= *1.0*	Mean=$4.39e^{1}$ SD=$1.802e^{1}$	Mean=$8.20e^{2}$ SD= $2.653e^{2}$	Mean=$9.53e^{1}$ SD=$8.339e^{1}$
$f_5(x)$ w= using (16)	Mean=$2.339e^{-42}$ SD=$9.426e^{-42}$	Mean=$1.866e^{-10}$ SD=$3.790e^{-10}$	Mean=$5.003e^{-28}$ SD=$4.370e^{-27}$
$f_5(x)$ w= *1.0*	Mean=$8.551e^{-28}$ SD=$6.951e^{-28}$	Mean=$2.0024e^{-05}$ SD=$5.849e^{-04}$	Mean=$7.452e^{-16}$ SD=$9.942e^{-16}$
$f_{13}(x)$ w= using (16)	Mean=0.0000 SD=0.0000	Mean=$6.046e^{-19}$ SD=$4.70e^{-18}$	Mean=$3.114e^{-32}$ SD=$1.990e^{-32}$
$f_{13}(x)$ w= *1.0*	Mean=$5.318e^{-15}$ SD=$8.927e^{-15}$	Mean=$9.884e^{-4}$ SD=$4.994e^{-4}$	Mean=$7.6042e^{-8}$ SD=$2.947e^{-8}$

c) Graphical analysis:

Figures 1-5 show a comparison between the standard GSA, PSO, and HGSPSO algorithm. The HGSPSO algorithm demonstrates better convergence than the standard version of both algorithms.

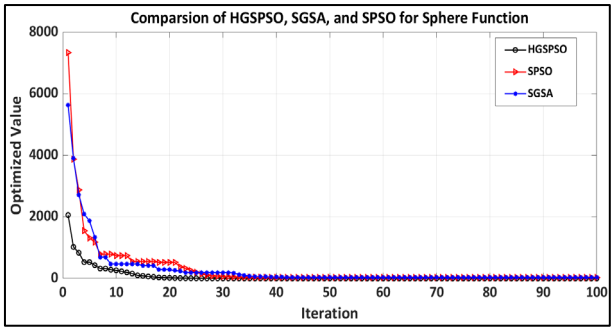

Figure 1 Comparison between HGSPSO and other standard versions of GSA and PSO algorithm for Sphere Function.

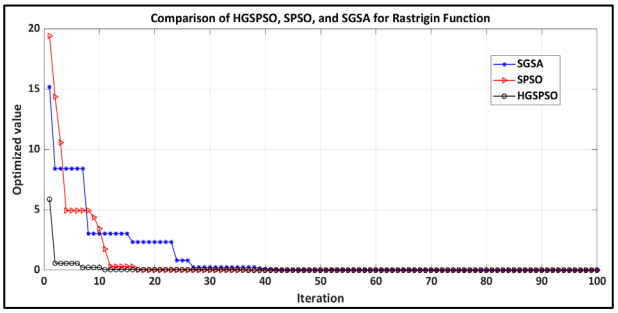

Figure 2 Comparison between HGSPSO and other standard versions of GSA and PSO algorithm for Rastrigin Function.

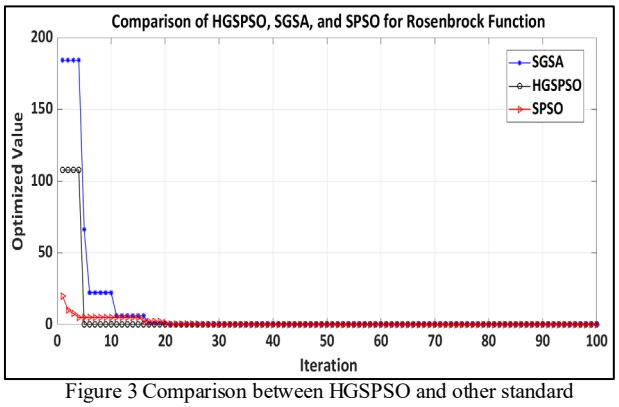

Figure 3 Comparison between HGSPSO and other standard versions of GSA and PSO algorithm for Rosenbrock Function.

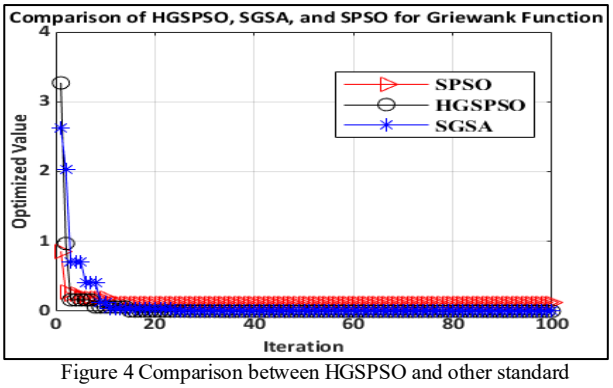

Figure 4 Comparison between HGSPSO and other standard versions of GSA and PSO algorithm for Griewank's Function.

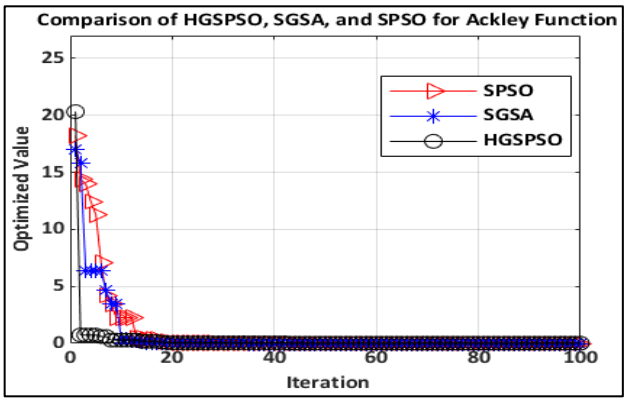

Figure 5 Comparison between HGSPSO and other standard versions of GSA and PSO algorithm for Ackley Function.

B. *Modern Benchmark Functions:*

In addition to five standard benchmark functions, the performance of the HGSPSO is also tested on some modern benchmark functions, which are taken from CEC 2017 [14]. The performance of the HGSPSO is compared with the following algorithms. These functions are defined in Table V.

1) *Comparison with State-of-the-Art New Methods:*

HGSPSO is compared with the following newly developed and modified versions of the existing methods are available in the literature. These algorithms are briefly described below.

a) LSHADE-RSP: Eugene et al. proposed a modified version of the L-SHADE algorithm based on a rank-based selective pressure strategy called LSHADE-RSP [15]. The primary changes in this variant are the modification in the mutation and crossover operators using selective pressure.

b) Hybrid Prey-Predator PSO (PP-PSO): A hybrid prey-predator PSO (PP-PSO) algorithm, based on the bio-inspired prey-predator relationship is proposed that used the three new approaches of breeding, escape and catch [16]. The inactive particles are changed or removed so the active particles will speed up the convergence.

c) jSO: Janez et al. introduced a modified version of the iL-SHADE algorithm that is called as jSO [17] in which the weighted version of the mutation strategy is improved. This algorithm stands at the second position in IEEE Congress on Evolutionary Computation (CEC) conference competition.

d) Teaching Learning Based Optimization with focused learning (TLBO-FL): Remya et al presented a modified version of the TLBO algorithm by introducing the student focus learning approach [18].

e) Improved Cuckoo Search (ICS)Algorithm: Rohit et al. proposed an improved variant of the Cuckoo search algorithm [19]. Three modifications have been made in this paper. Firstly, by reducing switch probability for maintaining a balance between global and local search. Introducing new equations for global and local searching are the second and third amendments.

Table V Modern Benchmark Functions

Category	*No*	*Functions*	$F_i^*=F_i=(x^*)$
Unimodal	$f_6(x)$	Shifted and Rotated Bent Cigar Function	100
	$f_7(x)$	Shifted and Rotated Sum of Different Power Function	200
	$f_8(x)$	Shifted and Rotated Zakharov Function	300
Multi-modal functions	$f_9(x)$	Shifted and Rotated Rosenbrock's Function	400
	$f_{10}(x)$	Shifted and Rotated Rastrigin's Function	500
	$f_{11}(x)$	Shifted and Rotated Expanded Scaffer's F6 Function	600
	$f_{12}(x)$	Shifted and Rotated Lunacek Bi_Rastrigin Function	700
	$f_{13}(x)$	Shifted and Rotated Levy Function	900
	$f_{14}(x)$	Shifted and Rotated Schwefel's Function	1000
Hybrid functions	$f_{15}(x)$	Hybrid Function 1 (N=3)	1100
	$f_{16}(x)$	Hybrid Function 5 (N=4)	1500
	$f_{17}(x)$	Hybrid Function 6 (N=5)	1700
	$f_{18}(x)$	Hybrid Function 6 (N=6)	2000
Composite functions	$f_{19}(x)$	Composition Function 1 (N=3)	2100
	$f_{20}(x)$	Composition Function 3 (N=4)	2300
	$f_{21}(x)$	Composition Function 5 (N=5)	2500
	$f_{22}(x)$	Composition Function 7 (N=6)	2700

2) Statistical Analysis

The parameters values of the compared techniques in table VII have been taken from the literature as sown in table VI.

Table VI Parameters of the compared algorithms

Algorithms	Parameters
LSHADE-RSP	$\mu F_r = 0.3$, $\mu Cr_r = 0.8$, $H = 5$, k=3, and $N_{max} = 75 \cdot D^{(2/3)}$
PP-PSO	Population size=*200*, $K_1 = 0.5$, $K_2 = 0.3$
jSO	$p_{max} = 0.25$; $p_{min} = p_{max}/2$, H = 5, M_F=0.3
TLBO-FL	Population size=100, T_F=1,
ICS	Initial population=50, $p_{max} = 0.25$; p_{min}=*0.8*

All these algorithms are compared with the HGSPSO in terms of t-test value, standard deviation, and mean. The results of the HGSPSO on these new benchmark test functions are compared with the algorithms and it clear from table VII that the HGSPSO gives an improved performance as compared to the other algorithms.

i) T-Test:

t-test criteria are the same as defined in section III-A.

TABLE VII. The standard deviation, mean and t-test value evaluation between HGSPSO and other algorithms for Modern Benchmark Functions for D= 30 (All Results Are Averaged (Rank: 1—Best, 6—Worst)

Functions		HGSPSO c_1=2.5-0.5 c_2=0.5-2.5	LSHADE-RSP	PP-PSO	jSO	TLBO-FL	ICS
$f_6(x)$	Mean	**0.0000**	0.0000	167	0.0000	$3.5e^{+3}$	$1.00e^{+10}$
	Std. Dev	**0.0000**	0.0000	$1.40e^{+02}$	0.0000	$3.6e^{+3}$	0.0000
	t-test	-	-	8.4348	-	6.8746	-
	Rank	***1***	1	2	1	3	4
$f_7(x)$	Mean	**0.0000**	0.0000	$4.88e^{+07}$	0.0000	$8.5e^{+16}$	$1.00e^{+10}$
	Std. Dev	**0.0000**	0.0000	$9.18e^{+07}$	0.0000	$5.8e^{+17}$	0.0000
	t-test	-	-	3.7589	-	1.0363	-
	Rank	***1***	1	2	1	4	3
$f_8(x)$	Mean	**0.0000**	0.0000	300	0.0000	$3.0e^{+03}$	$1.39e^{+02}$
	Std. Dev	**0.0000**	0.0000	$1.173e^{-03}$	0.0000	$1.1e^{+03}$	$9.30e^{+01}$
	t-test	-	-	$1.8085e^{+06}$	-	19.2847	10.5686
	Rank	***1***	1	3	1	4	2
$f_9(x)$	Mean	**$1.185e^{+1}$**	$5.8779e^{+01}$	411	$5.8670e^{+01}$	$9.0e^{+01}$	$1.43e^{+01}$
	Std. Dev	**$2.868e^{-2}$**	1.0784	$1.19e^{+01}$	$7.7797e^{-01}$	$2.4e^{+01}$	$2.18e^{+01}$
	t-test	-	307.6046	237.1772	425.3022	230.2352	7.9462
	Rank	***1***	4	6	3	5	2
$f_{10}(x)$	Mean	**7.3692**	7.5953	605	8.5568	$4.0e^{+01}$	$1.10e^{+02}$
	Std. Dev	**2.4225**	2.0252	$2.35e^{+01}$	2.0980	$2.1e^{+01}$	$2.51e^{+01}$
	t-test	-	0.5063	178.8771	2.6204	10.9150	28.7790
	Rank	***1***	2	6	3	4	5
$f_{11}(x)$	Mean	**$4.0892e^{-10}$**	$2.6838e^{-09}$	611	$6.0385e^{-09}$	$4.9e^{-01}$	$1.96e^{+01}$
	Std. Dev	**$9.5029e^{-10}$**	$1.8977e^{-08}$	3.56	$2.7122e^{-08}$	$4.2e^{-01}$	7.90
	t-test	-	0.8466	$1.2136e^{+03}$	1.4668	8.2496	17.5434
	Rank	***1***	2	6	3	4	5
$f_{12}(x)$	Mean	**$2.2803e^{+01}$**	$3.7959e^{+01}$	856	$3.8927e^{+1}$	$1.4e^{+02}$	$1.15e^{+02}$
	Std. Dev	**1.4394**	1.7138	$3.23e^{+01}$	1.4594	$4.7e^{+01}$	$2.06e^{+01}$
	t-test	-	47.8845	182.2214	55.6217	17.6238	31.5702
	Rank	***1***	2	6	3	5	4
$f_{13}(x)$	Mean	**0.0000**	0.0000	1404	0.0000	$3.4e^{+01}$	$1.95e^{+03}$
	Std. Dev	**0.0000**	0.0000	$2.32e^{+02}$	0.0000	$2.7e^{+01}$	$1.04e^{+03}$
	t-test	-	-	42.7922	-	8.9043	13.2583
	Rank	***1***	1	3	1	2	4
$f_{14}(x)$	Mean	**$1.1791e^{+03}$**	$1.4688e^{+03}$	4448	$1.5277e^{+03}$	$6.7e^{+03}$	$3.31e^{+03}$
	Std. Dev	**$2.6792e^{+02}$**	$3.0108e^{+02}$	$5.66e^{+02}$	$2.7716e^{+02}$	$2.8e^{+02}$	$2.68e^{+02}$
	t-test	-	5.0695	36.9120	6.3945	100.7366	39.7615
	Rank	***1***	2	5	3	6	4
$f_{15}(x)$	Mean	**3.0347**	3.0526	1249	3.0375	$8.2e^{+01}$	$4.20e^{+01}$
	Std. Dev	**4.4853**	8.5462	$5.16e^{+01}$	2.6464	$4.1e^{+01}$	$1.68e^{+01}$
	t-test	-	0.0131	170.1009	0.0038	13.5380	15.8454

	Rank	**1**	3	6	2	5	4
$f_{16}(x)$	Mean	1.0621	**7.2992e^{-01}**	1679	1.0879	2.2e^{+04}	3.41e^{+01}
	Std. Dev	9.493e^{-01}	**5.3154e^{-01}**	1.03e^{+02}	6.9133e^{-01}	2.3e^{+04}	8.39
	t-test	-	**-2.1589**	115.1875	0.1553	6.7633	27.6677
	Rank	*2*	***1***	5	3	6	4
$f_{17}(x)$	Mean	**2.1502e^{+01}**	3.1535e^{+01}	1961	3.2925e^{+01}	1.4e^{+02}	1.93e^{+02}
	Std. Dev	**6.5818**	6.8728	1. 37e^{+02}	8.0767	6.6e^{+01}	7.33e^{+01}
	t-test	**-**	7.4552	99.9892	7.7525	12.6329	16.4777
	Rank	***1***	2	6	3	4	5
$f_{18}(x)$	Mean	**2.3701e^{+01}**	2.8843e^{+01}	2269	2.9368e^{+01}	2.2e^{+02}	2.83e^{+02}
	Std. Dev	**3.9628**	6.8284	1.03e^{+02}	5.8548	1.2e^{+02}	8.51e^{+01}
	t-test	**-**	4.6054	154.0284	5.6680	11.5607	21.5222
	Rank	***1***	2	6	3	4	5
$f_{19}(x)$	Mean	**2.02639e^{+02}**	2.0758e^{+02}	2388	2.0929e^{+02}	2.3e^{+02}	2.50e^{+02}
	Std. Dev	**1.3727**	1.6596	2.70e^{+01}	1.9554	1.2e^{+01}	9.93e^{+01}
	t-test	**-**	16.2221	571.5890	19.6849	16.0182	3.3722
	Rank	***1***	2	6	3	4	5
$f_{20}(x)$	Mean	**3.1978e^{+02}**	3.5038e^{+02}	2787	3.5075e^{+02}	4.0e+^{02}	4.34e^{+02}
	Std. Dev	**3.1269**	3.3514	3.27e^{+01}	3.2992	1.6e^{+01}	5.30e^{+01}
	t-test	**-**	47.2063	531.0905	48.1783	34.7943	15.2124
	Rank	***1***	2	6	3	4	5
$f_{21}(x)$	Mean	**3.8171e^{+02}**	3.8670e^{+02}	2878	3.8670e^{+02}	4.0e^{+02}	3.83e^{+02}
	Std. Dev	**4.6180e^{-04}**	5.5921e^{-03}	4.07	7.6811e^{-03}	1.8e^{+01}	8.20e^{-01}
	t-test	**-**	6.2883e^{+03}	4.3370e^{+03}	4.5854e^{+03}	7.1850	11.1240
	Rank	***1***	3	6	4	5	2
$f_{22}(x)$	Mean	**4.51964e^{+02}**	4.9504e^{+02}	3187	4.9739e^{+02}	5.3e^{+02}	5.12e^{+02}
	Std. Dev	**6.1792**	6.9738	2.38e^{+01}	7.0017	2.1e^{+01}	9.32
	t-test	**-**	32.6903	786.5130	34.3966	25.2075	37.9633
	Rank	***1***	2	6	3	5	4
Overall Ranking (Average Ranking Number)		***1 (1.05)***	***2 (1.82)***	***6 (4.76)***	***3 (2.52)***	***5 (4.35)***	***4 (3.94)***

ii) Wilcoxon Signed-Rank Test:

The mean and standard deviation values of every method within the given function evaluations are used to assess the final solution's quality. Furthermore, the Wilcoxon signed-rank test at a 0.05 significance level (β) is used to analyze the dissimilarity between the two methods. Table VIII shows the results achieved by the Wilcoxon signed-rank test, which is based on the results of Table VII. Where *win*, *tie*, and *lose* show that HGSPSO, successes on w functions, draws on t functions, and fails on l functions than the other methods.

TABLE VIII results of Wilcoxon signed-rank test between HGSPSO and other methods on 30-D benchmark functions

HGSPSO v.s.	***LSHADE-RSP***	***PP-PSO***	**jSO**	***TLBO-FL***	**ICS**
w (+)	16	17	17	17	17
t (=)	0	0	0	0	0
l (-)	1	0	0	0	0

The HGSPSO results on the CEC 2017 benchmark functions are compared with its competitors. The CEC functions are more complex and difficult than the standard benchmark functions. These results are summarized in terms of mean, standard deviation, and t-test value in table V.

Simulations are performed to compare HGSPSO with other algorithms with D= 30 dimensions. Table VII shows that HGSPSO is the most successful method for solving the CEC functions in terms of the mean and standard deviation values. Also, the HGSPSO algorithm is most effective in searching better global optimal solutions particularly for finding out the theoretical global optimal at four functions *(f_{06}, f_{07}, f_{08}, and f_{13})* except for solving f_{16} function. HGPSO achieved superior accuracy in all functions. LSHADE-RSP algorithm stands at the second position for achieving better values followed by jSO, which is the third most effective algorithm, achieving improved solutions. The t-test value of the HGSPSO indicates that it has 95% superior performance than the rest of the methods

IV. DNA PROBLEM

In 1994, Adleman firstly introduced DNA computation. Deoxyribonucleic acid (DNA) is a nucleic acid that comprises the genetic instructions used in the growth and working of every living creatures and viruses [20].

Computation of the DNA depends mainly on the biochemical reactions of the DNA molecules which may cause an inappropriate result due to the poor quality of the DNA sequences [21]. DNA sequences used in DNA computation must fulfill different constraints that emphasis on the design of the DNA sequences that limit the probability of unwanted reactions [22]. Some of the previous work related to DNA computation done by various authors is briefly summarized in Table IX.

Table IX shows the previous work done in the literature on various objective functions and constraints

Objective Function	Objective functions used in the literature previously							
	[23]	[24]	[25]	[26]	[27]	[28]	[29]	[30]
Similarity (with Shift)	X	-	-	X	X	-	X	X
Similarity (without Shift)	X	-	-	X		X		X
H-measure (with Shift)	X	-	-	X	X	-	X	X
H-measure (without Shift)	X	-	-	X	X	-	X	X
Hairpin	X	-	-	X	X	-	X	-
GC-content	X	-	-	X	X	X	X	X
Free energy	-	X	X	-	-	-	-	-
Melting temperature	X	-	X	X	X	-	X	X
the occurrence of specific subsequences	-	-	-	-	-	X	-	X
Constraints on DNA bases	X	-	-	-	-	-	-	-
Secondary structure	X	-	X	-	-	X	-	-
Bio Lab Methods	-	X	-	-	-	-	-	-
Continuity	X	-	-	X	X	-	X	-
Reverse complement Hamming	-	-	-	-	-	X	-	-
3' end H measure	X	-	-	-	-	X	-	-

A. Elementary Concepts and descriptions

In this section, some of the elementary representations and relations are explained. $\Lambda = \{A, C, G, T, -\}$ is the representation of the nucleotide and a gap, each nucleotide is denoted by a single alphabet and gaps as "–". Similarly, if the gap is removed then nucleotide without a gap is denoted as $\Lambda nb = \{A, C, G, T\}$. For all the DNA sequences sets it is denoted by $\Lambda^* = \Lambda$ and Λ_{nb}. Suppose $a, b \in \Lambda\ a, b = \{A, C, G, T, -\}\ and\ x, y \in \Lambda^*\ x, y = \{A, C, G, T\}\ and\ \{A, C, G, T, -\}$. DNA sequence length is represented as $|x|$ where $x_i = (1 \le i \le |x|)$ that means i^{th} nucleotide from $5' - end$ of sequence *x*. A set of "n" number of sequences and length "l" is denoted by "Σ". similarly, for the i^{th} member, it is defined as Σ_i . Base "a" complementary is defined by $\bar{a}$. A few basic conditions and notations are defined as under. [31]

$$T(i,j) = \begin{cases} i, & i > j \\ 0, & otherwise \end{cases} \quad (31)$$

$$eq(a,b) = \begin{cases} 1, & a = b \\ 0, & otherwise \end{cases} \quad (32)$$

$$bp(a,b) = \begin{cases} 1, & a = b` \\ 0, & otherwise \end{cases} \quad (33)$$

In a given sequence the number of nonblank nucleotides $x \in \Lambda^*$ is defined as:

$$lenght_{nb}(x) = \sum_{i=1}^{|x|} nb(x_i) \quad (34)$$

$$where\ nb = nb(a) = \begin{cases} 1, & a \in \Lambda nb \\ 0, & otherwise \end{cases} \quad (35)$$

A shift of the sequence *x* by *i* bases is represented as

$$shift(x,y) = \begin{cases} (-)^i x_1 \dots x_{l-i}\ ,\ i \ge 0 \\ x_{i+1} \dots . x_l (-)^i,\ i < 0 \end{cases} \quad (36)$$

i) Continuity (f_1):

For calculating the recurrence of the same bases in a DNA sequence continuity is used. The DNA sequence might have an unexpected structure if the same bases are occurring repetitively.

$$f_{contiunity}(\Sigma) = \sum_{i=1}^{n} Contiunity\ (\Sigma_i) \quad (37)$$

$$Contiunity(x) = max_{1 \le i \le l}(\sum_{a \in \Lambda nb} c(a,i)) \quad (38)$$

$$c(a,i) = \begin{cases} n\ if\ ^\exists n, s.t. eq(a_i, a_{i+j}) = 1, for\ 1 \le j < n \\ eq(a_i, a_{i+n}) = 0, \\ 0 \quad otherwsie \end{cases} \quad (39)$$

ii) Hairpin (f_2):

Hairpin is used to calculate the likelihood to form a secondary structure. It is defined as under:

$$f_{Hairpin}(\Sigma) = \sum_{i=1}^{n} Hairpin\ (\Sigma_i) \quad (40)$$

$$Hairpin = \sum_{r=R_{min}}^{l-2\times p_{min}} \sum_{p=P_{min}}^{l-p_{min}-r} T\left(\sum_{i=1}^{pinlen(p,r)} bp(x_{P+1-i}, x_{p+r+i}), \frac{pinlen(p,r)}{2}\right) \quad (41)$$

$$pinlen\ (p,r,i) = \min(\ p\ +\ i, l\ -\ r\ -\ i\ -\ p)$$

$p = r = 6$ [32].

iii) H-measure (f_3):

H-measure is used to stop the cross-hybridization between the two sequences. It is represented as

$$f_{H-measure}(\Sigma) = \sum_{i=1}^{n} \sum_{j=1}^{n} H - measure(\Sigma_i, \Sigma_j) \quad (42)$$

where Σ_i and Σ_j are nonparallel to each other. It comprises of two parts. In the first term, which is known as the penalty, the term for the continuous complementary region, and the second part is called the overall complementarity. It is illustrated as under:

$$H - measure(x,y) = max_{g,i}(h_{dis}(x, shift(y(-)^g y, i)) + h_{con}(x, shift(y(-)^g y, i)) \quad (43)$$

Where $0 \le g \le l - 3\ and\ |i|\ \le l - 1$;

$$h_{con}(x,y) = \sum_{i=1}^{l} T(cbp(x,y,i), H_{con}) \quad (44)$$

$$h_{dis}(x,y) = T(\sum_{i=1}^{l} bp(x_i, y_i), H_{dis} \times lenght_{nb}(y)) \quad (45)$$

$$cbp(x,y,i) = \begin{cases} c\ \ if^\exists\ c, s.t. bp(x_i, y) = 0 \\ bp(x_{i+j}, y_{i+j}) = 1\ for 1 \le j \le c, \\ bp(x_{i+c+1}, y_{i+c+1}) = 0 \\ 0 \quad otherwise \end{cases} \quad (46)$$

iv) Similarity (f_4):

The main objective for calculating the similarity for two DNA sequences is to retain each of the sequence unique as possible.

$$f_{similarity}(\Sigma) = \sum_{i=1}^{n} \sum_{j=1}^{n} Similarity(\Sigma_i, \Sigma_j) \quad (47)$$

$$Similarity(x,y) = max_{g,i}(s_{dis}(x, shift(y(-)^g y, i)) + s_{con}(x, shift(y(-)^g y, i)) \quad (48)$$

Where $0 \le g \le l - 3\ and\ |i|\ \le l - 1$;

$$s_{con}(x,y) = \sum_{i=1}^{l} T(ceq(x,y,i), S_{con}) \quad (49)$$

$$s_{dis}(x,y) = T(\sum_{i=1}^{l} eq(x_i, y_i), S_{dis} \times lenght_{nb}(y)) \quad (50)$$

$$ceq(x,y,i) = \begin{Bmatrix} c \;\; if^{\ni} \; c, s.t. eq(x_i, y) = 0 \\ eq(x_{i+j}, y_{i+j}) = 1 \; for 1 \le j \le c, \\ eq(x_{i+c+1}, y_{i+c+1}) = 0 \\ 0 \quad otherwise \end{Bmatrix} \quad (51)$$

S_{con} is an integer between one and l and S_{dis} is a real value between zero and one. $T = 2$; $S_{dis} = H_{dis} = 0.17$ *and* $S_{con} = H_{con} = 6$ [32].

v) GC-content

$GC\text{-}_{content}$ is the percentage of G and C in a DNA sequence, for instance, if 10-mer DNA sequence GGTCCATCCA has two G's and four C's. (52) calculates $GC\text{-}_{content}$ as follows:

$$GC\text{-}_{content} = (X_G + X_C) / (X_A + X_T + X_G + X_C) \quad (52)$$

The $GC\text{-}_{content}$ of this DNA sequence is 60%.

vi) Melting Temperature (Tm)

For the experimental purpose, melting temperature "T_m" is the significant constraint to be controlled. The half of double-stranded DNA begins to break into its single-stranded form at this temperature [33].

B. Results and Analysis

Minimizing the objective functions is the key motive in this research. The fitness value of each objective function is calculated by applying the weighted function technique that transforms the multi-objective function into a single-objective function.

$$Optimize \; f_i(x)$$

$$\{ i \in \{h_{measure}, similarity, hairpin, continuity\}\}$$

$$subject \; to \; g_j(x) = 0, \; j \in \{T_m, \; GC_{content}\}$$

The fitness value is calculated as

$$Fitness \; value = \sum_i W_i \; f_i \; ; where \; W_i = 1$$

In the binary form A, C, G, and T are denoted as 0,1,2,3 in this paper. A DNA sequence entails various bases where each sequence represents one dimension. The DNA sequence length can be calculated as $(4^l - 1)$ which is equal to the boundary of the search area. For instance, the 20-mer sequence, the search area length is calculated as $(4^{20} - 1)$ that is from 0 to $1.0995116e^{+12}$ in a decimal number, in the same way, the boundary for $(4^3 - 1)$, i.e. from 0 to 63 and the sequence ranging from AAA to TTT. A three DNA sequence set can be obtained by using three dimensions. Therefore, each particle in the search area comprises of three DNA sequences. Since continuous HGSPSO is used as an alternative for the binary domain, therefore before converting all the decimal numbers into binary, the decimal numbers are rounded off to the nearest decimal value, then changed into binary, and then transformed into DNA sequences. For example, 403.8_{10} is rounded off to 404_{10} that is changed into binary form as 110010100_2, and DNA representation is "CGCCA". Table X highlights the sequences produced and the fitness values of the four objective functions.

The comparison results of the DNA codes produced by the HSPSO and other algorithms are presented in Table X. DNA codes produced by the HGSPSO gives better results than the other algorithms. The sequences generated by the proposed algorithm have lesser H-measure and similarity values. The obtained results indicate a greater possibility to hybridize with accurate complementary sequences through the HGSPSO. Additionally, the zero values of continuity and hairpin point towards the low probability of the secondary structure.

Figures 6-8 illustrate the pictorial representation of the mean values of all the DNA codes presented in table X for the three objective functions as the results for the continuity function is similar for all the methods

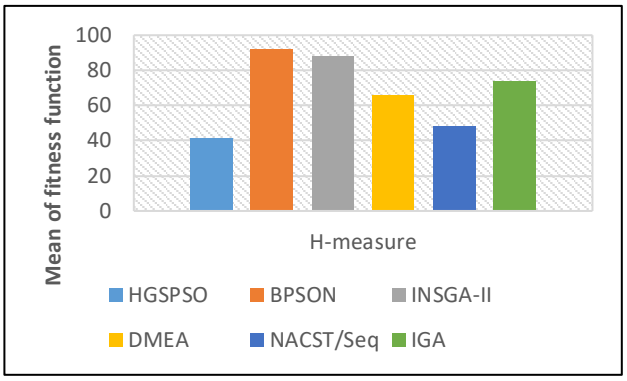

Figure 6 Comparison for the H-measure Objective function results between mean values of HGSPSO and its competitors

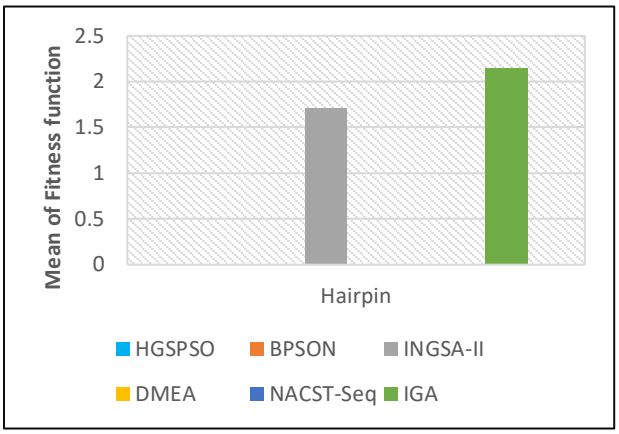

Figure 7 Comparison for the hairpin Objective function results between mean values of HGSPSO and its competitors

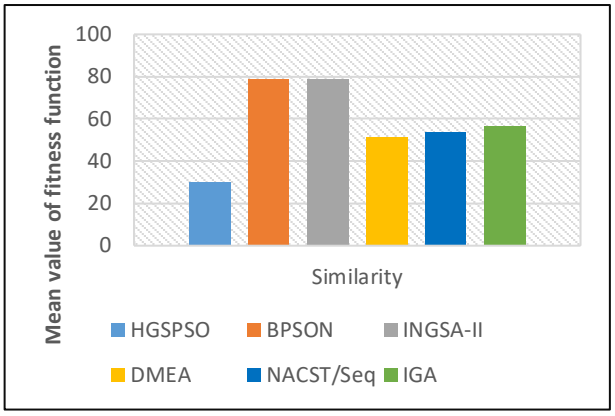

Figure 8 Comparison for the Similarity Objective function results between mean values of HGSPSO and its competitors

Table X Comparison of the sequences produced by HGSPSO and other algorithms in terms of objective functions

Sequences	Continuity	Hairpin	H-measure	Similarity	GC Content (%)
DNA Sequences produced by HGSPSO					
CCTAGTTCGTCCGTATTGTC	0	0	47	37	50
TTTATCGGTAGCGTGGTCTT	0	0	45	39	50
GGGTTCAATGCAGTGATCTG	0	0	44	35	50
ACTTGACGCTAGATGCCCTA	0	0	37	38	50
AAAGGCTGCTAGTCGATACC	9	3	41	34	50
CTAGGTTGCGGTAATCCATG	0	0	39	33	50
GATCATGTGATCGCAGTCCT	0	0	35	39	50
Mean	0	0	41.14285714	30.42857143	
Standard deviation	0	0	4.413183712	1.511857892	
DNA Sequences produced by BPSON [34]					
CCTTAGAACGTCTCGAGCTT	0	0	88	78	50
CCGTATACATGCTCGGTCTT	0	0	86	81	50
ACTCGCGACTACCAACATCT	0	0	98	75	50
CGTATCTGTGCTTCTCGTCA	0	0	92	84	50
ATTGCTGGATGAGGTGCCTA	0	0	94	80	50
TTCACTACTGAGGATCCGCA	0	0	98	76	50
CCTTAGAACGTCTCGAGCTT	0	0	88	78	50
Mean	0	0	92	78.85714286	
Standard deviation	0	0	4.898979486	3.078342164	
DNA Sequences produced by INSGA-II [35]					
CGTCTAGGCCGGATCAATAT	0	3	91	81	50
GGTTGTCCTGAGTGTTGTGT	0	0	84	80	50
AGAGTCAGCAGCGTAGAGAT	0	0	93	82	50
TACAGTCGGTTCGGTTATGG	0	0	90	73	50
GCGGAAGTAATCGGAAGTGA	0	0	88	81	50
ACGCCACAGTATATCATCGC	0	3	82	78	50
AGTCATTCTCCTGGCATTGC	0	6	90	76	50
Mean	0	1.714285714	88.28571429	78.71428571	
Standard deviation	0	2.360387377	3.946064948	3.251373336	
DNA Sequences produced by DMEA [31]					
GCCGGAGCCTTCTTGATAAT	0	0	68	53	50
AATCCTGCTTGTCCTCCTAC	0	0	63	50	50
TGAGCTCTCTGTTCCAACGA	0	0	64	52	50
ATGTAACACGCGGCCACTAA	0	0	63	50	50
ACTCGGATTGTGTTGAACGC	0	0	71	51	50
CGTTGTTGGCACCTACGTTA	0	0	68	54	50
ATCCAGACTACCAAGGCCAA	0	0	61	48	50
Mean	0	0	65.42857143	51.14285714	
Standard deviation	0	0	3.598941643	2.035400978	
DNA Sequences produced by NACST/Seq [36]					
CTCTTCATCCACCTCTTCTC	0	0	43	58	50
CTCTCATCTCTCCGTTCTTC	0	0	37	58	50
TATCCTGTGGTGTCCTTCCT	0	0	45	57	50
ATTCTGTTCCGTTGCGTGTC	0	0	52	56	50
TCTCTTACGTTGGTTGGCTG	0	0	51	53	50
GTATTCCAAGCGTCCGTGTT	0	0	55	49	50
AAACCTCCACCAACACACCA	0	0	55	43	50
Mean	0	0	48.28571429	53.42857143	
Standard deviation	0	0	6.799859943	5.623081683	
DNA Sequences produced by IGA [37]					
GTCGAAACCTGAGGTACAGA	9	3	72	58	50
GTGACTGTATGCACTCGAGA	0	3	74	57	50

ACGTAGCTCGAATCAGCACT	0	3	77	55	50
CGCTGATCTCAGTGCGTATA	0	0	74	55	50
GTCAGCCAATACGAGAGCTA	0	3	71	56	50
GCATGTCTAACTCAGTCGTC	0	3	68	55	50
TACTGACTCGAGTAGGACTC	0	0	78	60	50
Mean	1.28571428	2.142857143	73.42857143	56.57142857	
Standard deviation	3.40168025	1.463850109	3.457221565	1.902379462	

Table XI shows a comparison of the results summarized in terms of the mean and standard deviation values of all the objective functions. It is clear from the table that the weighted sum of the HGSPSO for all the functions is lesser than the other compared algorithm that demonstrates the better performance of the proposed method.

Table XI Results of the HGSPSO and its competitors in terms of mean and standard deviation

Functions		HGSPSO	BPSON	INSGAII	DMEA	NACST	IGA
f_1	μ	0	0	0	0	0	1.285
	σ	0	0	0	0	0	3.401
f_2	μ	0	0	1.71428	0	0	2.142
	σ	0	0	2.36038	0	0	1.463
f_3	μ	41.1428	92	88.2857	65.428	48.285	73.42
	σ	4.41318	4.8989	3.94606	3.5989	6.7998	3.457
f_4	μ	30.4285	78.857	78.7142	51.1428	53.428	56.57
	σ	1.51185	3.0783	3.25137	2.03540	5.6230	1.902
fitness value (weighted sum of all functions)		**71.85**	**170.8**	**168.7**	**116.57**	**101.7**	**133.4**

where "μ" represents mean and "σ" represents standard deviation.

V. CONCLUSIONS

In this paper, a hybrid algorithm is proposed through the fusion of the strong attributes GSA, PSO, and further modifying the velocity and position strategies. The primary concept behind this hybridization is to use the strong capacity of PSO in exploitation and the exploration capabilities of GSA. To validate the performance five standard and modern benchmark functions are used. The results highlight that the proposed method gives improved results for all the benchmark functions. It is also noticeable that the convergence of the HGSPSO is faster than PSO and GSA. Additionally, a DNA problem is also solved. HGSPSO outperformed the other algorithms in terms of the objective function results. It shows that the HGSPSO can translate DNA sequences with less probability of hybridization and greater precision.

Conflict of interest:

The corresponding author states that there is no conflict of interest.

Availability of data:

There is no data associated with this study.

REFERENCES

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

[2] M. Sabeti, R. Boostani, and B. Davoodi, "Improved particle swarm optimisation to estimate bone age," *IET Image Processing,* vol. 12, no. 2, pp. 179-187, 2018, doi: 10.1049/iet-ipr.2017.0545.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, Nov/Dec 1995 1995, vol. 4, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.

[4] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," *Information Sciences,* vol. 179, no. 13, pp. 2232-2248, 2009/06/13/ 2009, doi: https://doi.org/10.1016/j.ins.2009.03.004.

[5] J. Liang, B. Qu, and P. Suganthan, *Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization*. 2013.

[6] Y. Han, M. Li, and J. Liu, "Improving Hybrid Gravitational Search Algorithm for Adaptive Adjustment of Parameters," in *2017 13th International Conference on Computational Intelligence and Security (CIS)*, 15-18 Dec. 2017 2017, pp. 20-24, doi: 10.1109/CIS.2017.00013.

[7] N. Singh, S. Singh, and S. B. Singh, "A New Hybrid MGBPSO-GSA Variant for Improving Function Optimization Solution in Search Space," *Evolutionary Bioinformatics,* vol. 13, p. 1176934317699855, 2017, doi: 10.1177/1176934317699855.

[8] C. Li, S. Yang, and T. T. Nguyen, "A Self-Learning Particle Swarm Optimizer for Global Optimization Problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 42, no. 3, pp. 627-646, 2012.

[9] A. Zhang, G. Sun, J. Ren, X. Li, Z. Wang, and X. Jia, "A Dynamic Neighborhood Learning-Based Gravitational Search Algorithm," *IEEE Transactions on Cybernetics,* vol. 48, no. 1, pp. 436-447, 2018.

[10] A. A. Nagra, F. Han, Q. Ling, and S. Mehta, "An Improved Hybrid Method Combining Gravitational Search Algorithm With Dynamic Multi Swarm Particle Swarm Optimization," *IEEE Access,* vol. 7, pp. 50388-50399, 2019.

[11] K. Qian, W. Li, and W. Qian, "Hybrid Gravitational Search Algorithm Based on Fuzzy Logic," *IEEE Access,* vol. 5, pp. 24520-24532, 2017.

[12] F. Bergh and A. Engelbrecht, "A Convergence Proof for the Particle Swarm Optimiser," *Fundam. Inform.,* vol. 105, pp. 341-374, 01/01 2010, doi: 10.3233/FI-2010-370.

[13] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 4-9 May 1998 1998, pp. 69-73, doi: 10.1109/ICEC.1998.699146.

[14] M. Z. A. N. H. Awad, P. N. Suganthan, J. J. Liang, and B. Y. Qu, "Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization," in *IEEE Congress on Evolutionary Computation (CEC 2017)*, Donostia - San Sebastián, Spain, 05/06/2017 to 08/06/2017 2017.

[15] V. Stanovov, S. Akhmedova, and E. Semenkin, "LSHADE Algorithm with Rank-Based Selective Pressure Strategy for Solving CEC 2017 Benchmark Problems," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, 8-13 July 2018 2018, pp. 1-8, doi: 10.1109/CEC.2018.8477977.

[16] H. Zhang, M. Yuan, Y. Liang, and Q. Liao, "A novel particle swarm optimization based on prey-predator relationship," *Applied Soft Computing,* vol. 68, pp. 202-218, 2018/07/01/ 2018, doi: https://doi.org/10.1016/j.asoc.2018.04.008.

[17] J. Brest, M. S. Maučec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 5-8 June 2017 2017, pp. 1311-1318, doi: 10.1109/CEC.2017.7969456.

[18] R. Kommadath and P. Kotecha, "Teaching Learning Based Optimization with focused learning and its performance on CEC2017 functions," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 5-8 June 2017 2017, pp. 2397-2403, doi: 10.1109/CEC.2017.7969595.

[19] R. Salgotra, U. Singh, and S. Saha, "Improved Cuckoo Search with Better Search Capabilities for Solving CEC2017 Benchmark Problems," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, 8-13 July 2018 2018, pp. 1-7, doi: 10.1109/CEC.2018.8477655.

[20] C. Xu, Q. Zhang, B. Wang, and R. Zhang, "Research on the DNA Sequence Design Based on GA/PSO Algorithms," in *2008 2nd International Conference on Bioinformatics and Biomedical Engineering*, 16-18 May 2008 2008, pp. 816-819, doi: 10.1109/ICBBE.2008.200.

[21] Z. Ezziane, "DNA computing: applications and challenges," *Nanotechnology,* vol. 17, no. 2, pp. R27-R39, 2005/12/21 2005, doi: 10.1088/0957-4484/17/2/r01.

[22] C. Guangzhao and L. Xiaoguang, "The Optimization of DNA encodings based on Modified PSO/GA Algorithm," in *2010 International Conference On Computer Design and Applications*, 25-27 June 2010 2010, vol. 1, pp. V1-609-V1-614, doi: 10.1109/ICCDA.2010.5540892.

[23] F. Tanaka, M. Nakatsugawa, M. Yamamoto, T. Shiba, and A. Ohuchi1, "Developing Support System for Sequence Design in DNA Computing," Berlin, Heidelberg, 2002: Springer Berlin Heidelberg, in DNA Computing, pp. 129-137.

[24] R. Deaton, J. Chen, H. Bi, and J. Rose, *A Software Tool for Generating Non-crosshybridizing Libraries of DNA Oligonucleotides*. 2002, pp. 252-261.

[25] A. J. Hartemink, D. K. Gifford, and J. Khodor, "Automated constraint-based nucleotide sequence selection for DNA computation," *Biosystems,* vol. 52, no. 1, pp. 227-235, 1999/10/01/ 1999, doi: https://doi.org/10.1016/S0303-2647(99)00050-7.

[26] T. B. Kurniawan, N. K. Khalid, Z. Ibrahim, M. Khalid, and M. Middendorf, "An Ant Colony System for DNA sequence design based on thermodynamics," presented at the Proceedings of the Fourth IASTED International Conference on Advances in Computer Science and Technology, Langkawi, Malaysia, 2008.

[27] Z. Ibrahim, T. Kurniawan, N. Khalid, S. Sudin, and M. Khalid, "Implementation of ant colony system for DNA sequence optimization," *Artificial Life and Robotics,* vol. 14, pp. 293-296, 11/01 2009, doi: 10.1007/s10015-009-0683-0.

[28] U. Feldkamp, S. Saghafi, W. Banzhaf, and H. Rauhe, "DNASequenceGenerator: A Program for the Construction of DNA Sequences," Berlin, Heidelberg, 2002: Springer Berlin Heidelberg, in DNA Computing, pp. 23-32.

[29] S. Zhou, Q. Zhang, J. Zhao, and J. Li, "DNA Encodings Based on Multi-Objective Particle Swarm," *Journal of Computational and Theoretical Nanoscience,* vol. 4, pp. 1249-1252, 11/01 2007, doi: 10.1166/jctn.2007.005.

[30] M. Arita and S. Kobayashi, "DNA sequence design using templates," *New Gen. Comput.,* vol. 20, no. 3, pp. 263-277, 2002, doi: 10.1007/bf03037360.

[31] J. Xiao, X. Zhang, and J. Xu, "A membrane evolutionary algorithm for DNA sequence design in DNA computing," *Chinese Science Bulletin,* journal article vol. 57, no. 6, pp. 698-706, February 01 2012, doi: 10.1007/s11434-011-4928-7.

[32] S. Krishna Veni, M. S. Muhammad, S. M. W. Masra, Z. Ibrahim, and L. Kian Sheng, "DNA words based on an enhanced algorithm of multi-objective particle swarm optimization in a continuous search space," in *International Conference on Electrical, Control and Computer Engineering 2011 (InECCE)*, 21-22 June 2011 2011, pp. 154-159, doi: 10.1109/INECCE.2011.5953867.

[33] X. Zhang, Y. Wang, G. Cui, Y. Niu, and J. Xu, "Application of a novel IWO to the design of

encoding sequences for DNA computing," *Computers & Mathematics with Applications,* vol. 57, no. 11, pp. 2001-2008, 2009/06/01/ 2009, doi: https://doi.org/10.1016/j.camwa.2008.10.038.

[34] K. Liu, B. Wang, H. Lv, X. Wei, and Q. Zhang, "A BPSON Algorithm Applied to DNA Codes Design," *IEEE Access,* vol. 7, pp. 88811-88821, 2019, doi: 10.1109/ACCESS.2019.2924708.

[35] Y. Wang, Y. Shen, X. Zhang, G. Cui, and J. Sun, "An Improved Non-dominated Sorting Genetic Algorithm-II (INSGA-II) applied to the design of DNA codewords," *Mathematics and Computers in Simulation,* vol. 151, pp. 131-139, 2018/09/01/ 2018, doi: https://doi.org/10.1016/j.matcom.2018.03.011.

[36] S. Soo-Yong, L. In-Hee, K. Dongmin, and Z. Byoung-Tak, "Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing," *IEEE Transactions on Evolutionary Computation,* vol. 9, no. 2, pp. 143-158, 2005, doi: 10.1109/TEVC.2005.844166.

[37] B. Wang, Q. Zhang*, and R. Zhang, "Design of DNA Sequence Based on Improved Genetic Algorithm," Berlin, Heidelberg, 2008: Springer Berlin Heidelberg, in Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues, pp. 9-14.