







A Strategy-Based Optimization Algorithm to Design Codes for DNA Data Storage System

Abdur Rasool, Qiang Qu, Qingshan Jiang and Yang Wang

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

December 4, 2021

A Strategy-based Optimization Algorithm to Design Codes for DNA Data Storage System*

Abdur Rasool^{1,2} , Qinag Qu^{1,*} , Qingshan Jiang¹ , and Yang Wang¹ 

¹ Shenzhen Key Laboratory for High Performance Data Mining, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

*Correspondence: qiang@siat.ac.cn

² Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Beijing 100049, China
rasool@siat.ac.cn

Abstract. The exponential increase of big data volumes demands a large capacity and high-density storage. Deoxyribonucleic acid (DNA) has recently emerged as a new research trend for data storage in various studies due to its high capacity and durability, where primers and address sequences played a vital role. However, it is a critical biocomputing task to design DNA strands without errors. In the DNA synthesis and sequencing process, each nucleotide is repeated, which is prone to errors during the hybridization reactions. It decreases the lower bounds of DNA coding sets which causes the data storage stability. This study proposes a metaheuristic algorithm to improve the lower bounds of DNA data storage. The proposed algorithm is inspired by a moth-flame optimizer (MFO), which has exploration and exploitation capability in one dimension, and it is enhanced by opposition-based learning (OBL) strategy with three-dimension search space for the optimal solution; hereafter, it is MFOL algorithm. This algorithm is programmed to construct the DNA storage codes by reducing the error rates of DNA coding sets with GC-content, Hamming distance, and No-runlength constraints. In experiments, 13 benchmark functions and Wilcoxon rank-sum test are implemented, and performances are compared with the original MFO and three other algorithms. The generated DNA codewords by MFOL are compared with a state-of-the-art Altruistic algorithm and KMVO algorithm. The proposed algorithm improved 30% DNA coding rates with shorter sequences, reducing errors during DNA synthesis and sequencing.

Keywords: DNA data storage · Biocomputing · DNA coding sets · Opposition-based learning · MFO algorithm

1 Introduction

International Data Corporation estimated that the digital data would exponentially grow from 33 ZB to 175 ZB (2.5 EB/day) during 2018-2025 due to ex-

* Supported by The National Key Research and Development Program of China under grant number 2020YFA0909100.

tensive usage of IoT worldwide [1]. Meanwhile, the limitation of storage density and longevity in the existing storage media demands the development of the latest technology. DNA is a step-forward molecular-based solution due to primers that play a vital role in its density and long-lasting stability. It comprises four nucleotides – adenine (A), thymine (T), cytosine (C), and guanine (G). The A and T nucleotides are integrated by dual Hydrogen bonds while C and G with triple H-bond and form a double helix with the pairing of complementary bases known as hybridization. In DNA synthesis, primers are added into the strands during the data writing which is utilized in polymerase chain reaction (PCR) amplification for particular required data during the data reading process. As DNA molecules lack particular spatial organization, the encoded strands should have a specific address to recognize their location in the data stream [2]. Each DNA strand is divided into blocks to encode a big amount of data, as presented in Fig. 1. DNA has the capability to store 4.2×10^{21} bits binary data per gram of single-strand, which is 420 billion times high-performance bio-computing than existing electromagnetic media [3]. DNA data storage technology can be defined by following three fundamental steps [3–5]: (1) Digital data is converted into binary form and encoded into DNA strands (A, T, C, or G strings) with a particular coding scheme. These strings are called DNA codes or codewords. (2) The encoded DNA strands are synthesized into oligonucleotides by a DNA synthesizer, and data is stored. This process is called data writing on DNA. (3) DNA strands are decoded by DNA sequencing to retrieve the original digital data, which is called data reading from DNA.

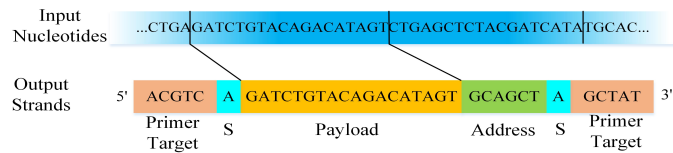


Fig. 1. DNA strand structure with primers, payload, sense (s) and address.

A plethora of studies have shown the novel tracks for its developments. For instance, Church [4] encoded 5.27 MB files into DNA chemical molecules and efficiently decoded those files by DNA sequencing. Goldman [5] proposed a scalable method to store the 739 kilobytes of digital information. Bornholt [3] delivered random access features by XOR encoding scheme and synthesized DNA of 151 KB dataset. The author has developed an end-to-end DNA data storage system to overcome the challenges of high risks in data loss [6]. It proposed a self-contained DNA storage system with three different methods, which thoroughly reduce the data redundancy and improve the DNA coding sets. The reported results indicate the significance of DNA sequence codes. All the above-mentioned studies found that DNA coding sets directly affect DNA synthesis and sequences efficiency. Thus, it is strongly required to develop robust DNA coding sets that

must satisfy the DNA coding constraints. It is compulsory to detect the error source to avoid the insertion, substitution, and deletion errors that occur during the development of DNA codes. For example, a DNA code: ACAGGGTACT, G has been consecutively repeated, which will be considered a single G for the reading process and caused the lowest convergence rate for the DNA reading and writing. Thus, GC content and homopolymer length constraints are deemed initially in the DNA Fountain code [2].

Song [7] proposed a coding method that satisfied No-runlength and GC-content constraints, but still, it generated errors during the encoding and decoding and reduced the DNA codes. Therefore, DNA codewords have been significantly improved with the constraints and stochastic search algorithm recently. Metaheuristic algorithms, i.e., moth-flame optimization (MFO) [8], firefly optimization algorithm (FOA) [9], grey wolf optimizer (GWO) [10], harris hawks optimization (HHO) [11], and mean-variance optimization (MVO) [12], have been efficiently used in various aspects of computation engineering. For example, KMVO [13] has been adopted for DNA coding sets for DNA-based data storage by using MVO. Their results stated that the proposed algorithm attained a bunch of sequences that satisfy the energy-free constraint at a particular melting temperature. The motivations of this paper are KMVO and Altruistic algorithms [14]. KMVO algorithm attained the 1.5 times higher DNA coding set than Altruistic algorithm with the same purpose. However, authors [13, 14] suggested it further can be improved by mutation strategy. In biocomputing, A mutation strategy enables the alteration in the nucleotides sequence to generate high-quality codes—for example, the opposition-based learning (OBL) strategy focuses on exploring the solution in the opposite dimension. To the best of our knowledge, the MFO algorithm with the OBL mutation strategy has not been reported in the literature to construct DNA coding sets for data storage.

In this paper, moth-flame optimization (MFO) has been enhanced by the opposition-based learning (OBL) strategy to generate DNA coding sets – hereafter, its MFOL. MFO algorithm has the exploration and exploitation ability in one dimension of search space that is improved by OBL; which considers the opposite solution of the concerned solution in three dimensions to boost the local search capability. The experiments are conducted on 13 benchmark functions, including unimodal and multimodal functions. As a result, the MFOL algorithm efficiently enhanced the global exploration and exploitation abilities to improve the convergence rates. The results are compared with 4 different well-known optimizers. MFOL algorithm is applied to design DNA codes for DNA-based data storage systems. To overcome the critical issue of error occurrence during the DNA synthesis, the MFOL algorithm is utilized with GC-content, Hamming distance, No-runlength constraints. The results are compared with the state-of-the-art Altruistic algorithm and KMVO. The overall mechanism of the MFOL algorithm with the opposition-based learning strategy and existing constraints reported the high quality of large DNA coding sets with improved lower bounds to construct a dense-based DNA storage system. The significant contributions are as follows:

- A novel algorithm (MFOL) is proposed based on a moth-flame optimizer which synergy by opposition-based learning mutation strategy for faster convergence and stronger exploration and exploitation capabilities.
- The proposed algorithm is applied to construct DNA codewords. It contributes to improving the lower bounds of DNA coding sets, and it is validated by computing the temperature variance.
- The MFOL algorithm satisfies the DNA coding constraints to avoid the non-specific hybridization for storing larger digital data files in the shorter sequence of DNA to deliver a stable data storage system.

The structure of the rest article is as follows; Section II introduces the existing DNA constraints, Section III elaborates the proposed algorithm, Section IV explains the experiments and results, Section V concludes the work.

2 DNA Code Constraints

The critical aspect of DNA storage is to design DNA strands with the least errors during its vital processes; synthesis and sequencing. The existing state-of-the-art constraints (GC-content, Hamming distance, and No-runlength) are used to design the DNA codes with $C(n, M, d)$, wherein n is the length of the sequence and d presents Hamming distance and M is a symbol to indicate GC-content with $\lfloor n/2 \rfloor$ parameters.

The GC-content constraint in $C(n, M, d)$ set is the ratio of the sum of bases content (G and C) to the total number of bases. It can be defined for s sequence length as [7]:

$$GC(s) = \frac{|G + C|}{|s|} \times 100\% \quad (1)$$

Similarly, the No-runlength constraint is to avoid the existence of homopolymers in a sequence, and for a DNA sequence S with n bases ($S_1, S_2, S_3, \dots, S_n$) can be presented as [11]:

$$S_i \neq S_{i+1}, i \in [1, n - 1] \quad (2)$$

Meanwhile, the Hamming distance H between 2 sequences (α & β) of the same length in $C(n, M, d)$ set can be computed by the sum of different base elements by satisfying the $H(\alpha, \beta) \geq d$, where, d is threshold [15]:

$$H(\alpha, \beta) = \sum_{i=1}^n h(\alpha_i, \beta_i), h(\alpha_i, \beta_i) = \begin{cases} 0, & \alpha_i = \beta_i \\ 1, & \alpha_i \neq \beta_i \end{cases} \quad (3)$$

In Hamming distance, H determines the similarity between 2 DNA sequences by calculating the d . The greater H shows the greater distance and less similarity among the particular sequences, which avoids non-specific hybridization. Overall, these constraints aim to develop feasible DNA codes with different lengths.

3 Proposed Algorithm – MFOL

The proposed algorithm leverages the moth-flame optimizer (MFO) [8], where the moth uses a transverse orientation (TO) navigation mechanism. The TO method enables a moth to fly by adjusting a fixed angle by the moon’s focal point. Meanwhile, the moth collided with artificial lights and lost its destination. However, the moth persists in maintaining the same angle, which causes its deadly spiral path. This concept provides a mathematical optimizer algorithm that supports the convergence of an object or moth. MFO algorithm has two candidate solutions; Moths (M) and Flame (F). In a population-based algorithm, there can assume another array of fitness (f) values for all solutions. Both candidate solutions can be considered in the following matrices.

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} & \dots & m_{1,d} \\ m_{2,1} & m_{2,2} & \dots & m_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ m_{n,1} & m_{n,2} & \dots & m_{n,d} \end{bmatrix}, \quad Mf = \begin{bmatrix} Mf_1 \\ Mf_2 \\ \vdots \\ Mf_n \end{bmatrix} \quad (4)$$

$$F = \begin{bmatrix} F_{1,1} & F_{1,2} & \dots & F_{1,d} \\ F_{2,1} & F_{2,2} & \dots & F_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ F_{n,1} & F_{n,2} & \dots & F_{n,d} \end{bmatrix}, \quad Ff = \begin{bmatrix} Ff_1 \\ Ff_2 \\ \vdots \\ Ff_n \end{bmatrix} \quad (5)$$

where n shows the candidate solution number and d is the dimension variable.

The only difference between both solutions is the system how we deal with them in the iteration process. The moth flies in the search space and acts as a search agent, while the flame is the optimal solution for the moth to achieve it as a destination in the search space. Thus, a moth flies around the search space by focusing the destination (flame) on finding a globally optimal solution. In this paper, the parameters have been chosen as given in the original work of MFO. As the motivation of this optimizer is TO, the moth updates its position corresponding to the flame with the following mathematically model:

$$M_i = S(M_i, F_j) = D_i \cdot e^{bt} \cdot \cos(2\pi t) + F_j \quad (6)$$

where M_i represents the i -th moth, S indicates the spiral function, F_j presents the j -th flame, D_i shows the distance of i -th moth for the j -th flame, b is a constant for spiral function, and t is a random number $[-1, 1]$.

Eq. (6) adjust the moth’s spiral path, which allows a moth to fly around a flame. The spiral function is a key component of MFO which decides the moth movement with respect to flames. Thus, it enables the MFO algorithm to attain the ability of exploration and exploitation in the search space. The logarithmic spiral, position with different t curves and space around the flame are illustrated in Fig. 2.

Apart from these spiral functions of the MFO algorithm, the following variable-based array is also considered as lower bounds for MFO:

$$ub = [ub_1, ub_2, ub_3, \dots, ub_{n-1}, ub_n] \quad (7)$$

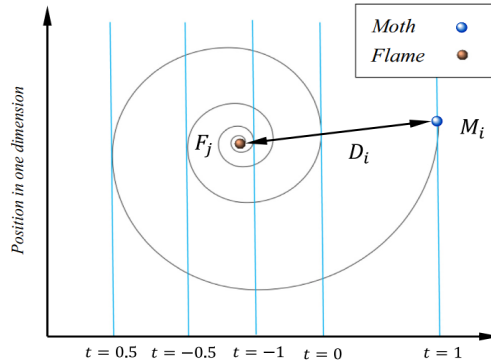


Fig. 2. Logarithmic spiral, position concerning t and space around the flame [8].

$$lb = [lb_1, lb_2, lb_3, \dots, lb_{n-1}, lb_n] \quad (8)$$

where ub and lb indicate the upper bounds and lower bounds with n number of moths, respectively.

These bounds decide the search space limit of the moth after the initialization. The optimization of this mechanism enables the moth to acquire the best position in the search space. However, a problem can occur due to one dimension search space that causes MFO to fall into local optima and affects searchability. In order to maintain the balance between exploitation and exploration and find the best optimal solution, this paper utilized the following mutation strategy.

3.1 Opposition-based Learning Strategy

In optimizing any problem, solution Z is estimated as \check{Z} , which is not the exact solution. It is not a best practice to consider the initial guess as to the best result. Practically, for all the optimal solutions, the optimized system should focus on all dimensions or aspects, more specifically toward the opposite direction/dimension [16]. Tishoosh et al. (2005) reported opposition-based learning (OBL) mutation strategy for computational intelligence [17]. The OBL strategy tackles the moth solution Z in three dimensions (3D) if its searching is advantageous in the opposite direction with opposite moth solution \check{Z} . In which, considering the 3D interval (a, b, c) , the solution for the concern problem can be observed in moth Z . The \check{Z} will be generated at the opposite interval (a', b', c') of initial moth Z , as illustrated in Fig. 3. It will prior search the opposite moth solution \check{Z} according to the following definitions.

Definition I: Let $Z \in R$ is a real number for a particular interval; $Z \in [a, b, c]$. The opposite number \check{Z} can be defined as follows:

$$\check{Z} = a + b + c - Z \quad (9)$$

For $a + b = 0$ and $c = 1$, or vice versa, then,

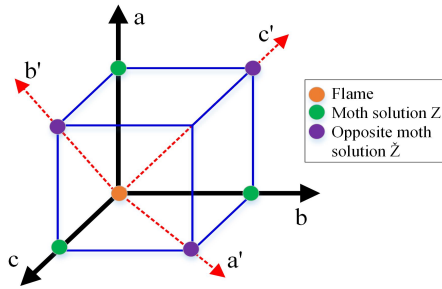


Fig. 3. Three-dimension search space for the moth solution Z and opposite moth solution \check{Z} for the Opposition-based learning mutation strategy.

$$\check{Z} = 1 - Z \quad (10)$$

Meanwhile, this behavior of opposite number for multiple dimensions can be defined as follows:

Definition II: let $Z(a_1, a_2, \dots, a_n)$ is a location in n -dimensional search space for the coordinate system with $a_i \in [x_i, y_i]$ and $i \in 1, 2, \dots, n$. The opposite location or area $\check{Z}_i = (a'_1, a'_2, \dots, a'_n)$ is defined as follows:

$$\check{Z}_i = a'_1 + a'_2 + a'_n - Z_i \quad i = 1, 2, \dots, n. \quad (11)$$

Based on Eq. (11), the moth Z or opposite moth \check{Z} are close to a solution with respect to the flame. The 3-dimension interval can recursively optimize until either moth or opposite moth come close enough to the targeted solution. These characteristics furnish the opportunity for the MFOL algorithm to access the global optima solution by balancing between exploitation and exploration abilities. The computation time complexity of MFOL is also same as the MFO algorithm ($O(MFOL) = O(tn^2 + tnd)$), where d is the number of variables and t is the maximum number of iterations [8]. The pseudo-code of the MFOL algorithm is presented in Algorithm 1. The architecture of MFOL is illustrated in Fig. 4.

4 Experiments & Results Evaluation

The experiments were executed in an integrated environment; for instance, all algorithms performed on MacBook 2.4 GHz, 8 GB DDR3, Python with 3.7.10v, platform Google's Colab, and 3D convergence plots into MATLAB R2018b. To construct DNA codewords, DNA bases (A, T, C, and G) are mapped with the quarterly number (A-0, T-1, C-2, and G-3). It employed 13 mainstream functions (mathematically defined in [8]) to demonstrate the optimization performance of MFOL. A set of different parameters have been implemented. However, the significant results presented in this paper are based on these parameters; the number of moths or population size: 50, and the number of iterations for each

Algorithm 1: Pseudocode of proposed MFOL algorithm

Input: The population size N for two candidate solutions (M, F) , Location of moth (L), FitnessFunction of moth M_f , Fitness Function of flame N_f .

Output: Global best individual solution X_m .

```

1: Initialize random population  $X_i$ 
2: for (each moth  $X_i$ ) do
3: Calculate fitness of  $M_f$  and  $F_f$  population using Eq. (4) and (5);
4: if (population  $N$  converge) then
5:   Update the moths' position  $L$  for  $lb$  using Eq. (8);
6:   Compute global optimal  $L$  with opposition-based learning strategies
   (Eq. 11); else
7:   while (not converge) do
8:     for  $i = 1 : n$ ;
9:       update candidate solutions  $(M, F)$  with Eq. (6)
10:    end for
11:   end while end if
12: end for
Return: Global optimal solution  $X_m$ .

```

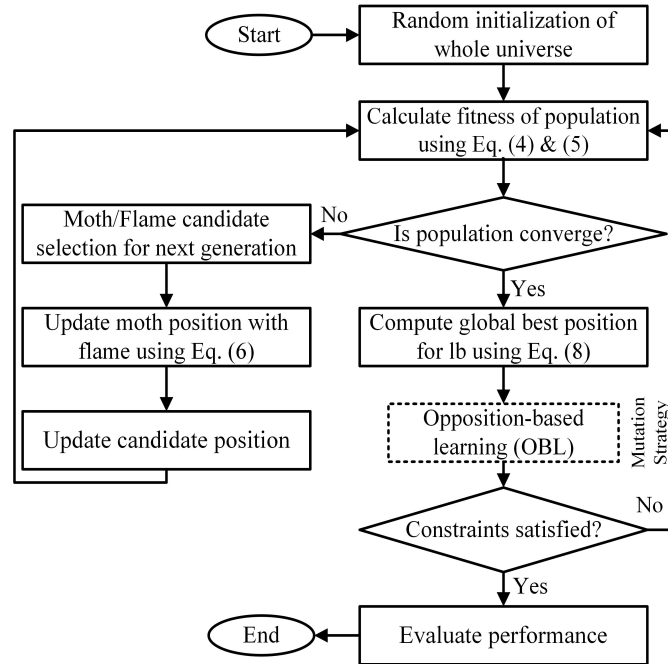


Fig. 4. An Architecture of the proposed MFOL algorithm is presented based on the MFO algorithm and Opposition-based learning mutation strategy.

function: 500. If an algorithm performs for n times will yield average or standard deviation (SD) with the best solution. The following mechanism is utilized to report the optimal solution for average and SD values.

- The lowest the average value, the highest the algorithm' performance.
- The minimum the SD value, the maximum the stability of the algorithm.

Additionally, the proposed algorithm MFOL is compared with the original MFO [8] and the other three algorithms; FOA [9], GWO [10], and HHO [11]. A non-parametric Wilcoxon Rank-sum test [18,19] is accompanied to validate the result's originality of MFOL algorithms and compare with MFO [8]. Furthermore, the MFOL algorithm is trained with available DNA coding constraints, i.e., GC-content, Hamming distance, etc., to overcome the occurrence of the error of sequences for the DNA storage effectiveness. The lower bounds values are compared with the state-of-the-art Altruistic algorithm [14]. Eventually, a thermodynamic analysis is performed on existing constraints to validate the generated sequences by computing the temperature variance of DNA coding sets.

4.1 Benchmark Functions' Evaluation

This study used 2 types of benchmark functions; Unimodal (F1-F7) and Multimodal Functions (F8-F13), to test the MFOL performance. Unimodal functions have the exploitation capability, deal only with 1 global optimal score, and do not consider the local optimal values. In contrast, multimodal functions have exploration ability due to having numerous numbers of local optimal solutions [8].

Table 1. Comparison of different algorithms with MFOL for Unimodal functions.

Functions	Metrics	MFO [8]	FOA [9]	GWO [10]	HHO [11]	MFOL
F1	AVG	8.63E+03	3.61E+03	6.29E+02	1.51E+04	1.49E+02
	SD	1.48E+04	9.78E+03	4.80E+03	2.14E+04	3.80E+03
F2	AVG	-7.61E+03	-3.55E+03	-4.18E+03	-1.22E+04	0.00E+00
	SD	1.26E+03	2.22E+02	1.91E-02	1.02E+03	0.00E+00
F3	AVG	3.50E+04	1.75E+04	3.24E+03	4.77E+04	0.00E+00
	SD	1.65E+04	2.52E+04	1.28E+04	3.53E+04	0.00E+00
F4	AVG	6.44E+00	1.38E+01	3.51E+00	5.62E+01	0.00E+00
	SD	6.55E+00	2.26E+01	1.29E+01	9.36E+00	0.00E+00
F5	AVG	2.01E+07	7.58E+06	1.63E+06	9.15E+07	0.00E+00
	SD	4.94E+07	2.76E+07	1.60E+07	1.04E+08	0.00E+00
F6	AVG	3.44E+01	5.04E+04	5.59E+04	2.36E+04	5.27E+01
	SD	1.26E+04	5.10E+03	2.64E+03	6.78E+03	3.05E+03
F7	AVG	1.55E+01	4.75E+00	7.31E-01	3.57E+01	1.17E+00
	SD	2.04E+01	1.41E+01	7.04E+00	4.50E+01	4.10E-01

Tables 1 and 2 indicate the average and standard deviation of unimodal and multimodal functions, respectively. In Table 1, a general trend presents the improved performance of our proposed algorithm in various functions. For instance, the functions F2-F5 achieved the best convergence in both matrices of average and SD. However, the score of MFOL with F6 lags behind the original MFO due to probably larger optimization intervals. In table 2, MFOL exhibits superior performance with the lowest average and SD scores as compared to MFO [8]. The average and SD scores of F8 and F10-F12 secured the global optimal solution after 500 iterations, demonstrating the proposed algorithm jumping-out performance from the local optimum. Meanwhile, the proposed algorithm failed to attain the maximum global optimal solution for F9 due to high variance values as compared to the rest of the other algorithms. This insufficient result may be appeared due to the moth's large interval for optimization in search space. As compared to the remaining three optimizers, the proposed optimizer competitively jumps out of the local optimum and secures itself in the global optimum solution with minimum magnitude and variances. These significant results indicate the demand and importance of the OBL strategy.

Table 2. Comparison of different algorithms with MFOL for Multimodal functions.

Functions	Metrics	MFO [8]	FOA [9]	GWO [10]	HHO [11]	MFOL
F8	AVG	1.09E+04	3.33E+10	2.81E+10	8.32E+09	0.00E+00
	SD	1.18E+11	7.43E+11	6.28E+11	1.82E+11	0.00E+00
F9	AVG	1.92E+02	1.64E+02	1.50E+02	1.60E+02	2.42E+02
	SD	6.75E+01	9.48E+01	8.45E+01	1.26E+02	5.93E+01
F10	AVG	1.69E+01	8.56E+00	1.65E+01	1.19E+01	6.16E-01
	SD	1.70E+00	3.99E+00	2.75E+00	2.88E+00	8.28E-01
F11	AVG	9.18E+01	1.12E+00	5.71E+00	1.40E+02	1.05E+00
	SD	1.34E+02	8.87E+01	4.35E+01	2.02E+02	1.19E+00
F12	AVG	3.50E+07	1.34E+07	3.43E+06	2.57E+08	2.61E+00
	SD	1.01E+08	5.70E+07	3.56E+07	2.76E+08	3.99E+00
F13	AVG	1.00E+08	2.88E+07	7.02E+06	4.23E+08	2.38E+04
	SD	1.98E+08	1.16E+08	7.09E+07	4.89E+08	7.37E+08

4.2 Convergence Efficiency

The convergence curve is a vital criterion to assess the algorithm convergence speed and capability to jump out from the local optima [8]. The convergence curves with 3D representation are depicted in Figure 5. This paper considers only one function's outcome from each unimodal and multimodal function due to their significant convergence efficiency and paper page limit. In the unimodal

F5 function, MFOL converges speedily than MFO and other algorithms to attain the global optimal solution. In contrast, in the multimodal F12 function, the proposed algorithm achieved optimal solution at 50 iterations, while MFO fell into the local optima. In summary, the MFOL convergence curves are experimentally guaranteed by quantitative and qualitative metrics that exhibit the competitive results over the state-of-the-art algorithms by establishing a balanced nature between exploration and exploitation.

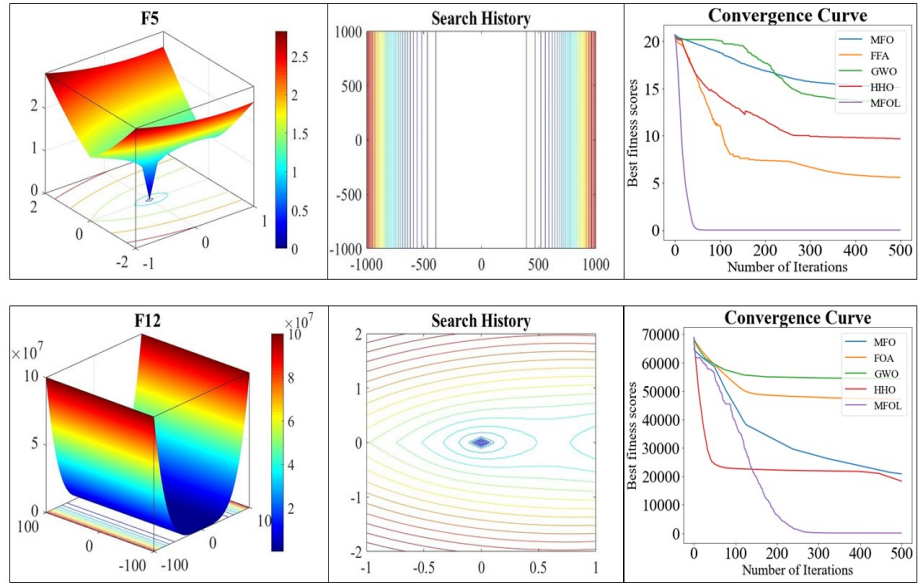


Fig. 5. A 3D representation and convergence efficiency of Unimodal function F5 and Multimodal function F12 is illustrated separately. Each function has its particular search space, search history, and convergence curves indicating the highest performance for the MFOL algorithm.

4.3 Wilcoxon Rank-sum Test

The results of benchmark functions indicate the algorithm’s general performance, while the statistical test, Wilcoxon Rank-sum Test, proves the algorithm’s statistical significance [19]. According to the Wilcoxon test’s hypothesis, an algorithm is considered statistically significant if the *P* – value is greater than 0.05. This study compared the statistical significance between MFO and MFOL by ranking any 2 samples of the 30 iterations from unimodal and multimodal functions. In Fig. 6, the results met the criteria in most cases which present the optimum results of the proposed algorithm.

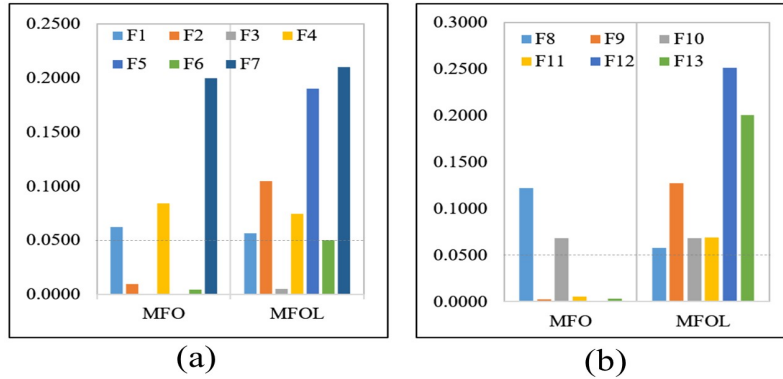


Fig. 6. Comparison of Wilcoxon rank-sum test for MFO [8] and MFOL with (a) Unimodal functions and (b) Multimodal functions. A dotted line at the P -value of 0.0500 indicates the threshold level for this rank-sum test.

4.4 Bounds on DNA Storage Constraints Coding

MFOL algorithm is trained and practically applied to improve the lower bounds of DNA storage coding sets with GC-content, No-runlength, and Hamming distance constraints $C^{GC,NL}(n, M, d)$, where n indicates sequence length, d presents the Hamming distance, and M shows the GC-content with $\lfloor n/2 \rfloor$ parameters. The Altruistic [14] and KMVO [13] algorithms used $4 \leq n \leq 10$ and $3 \leq d \leq n$ bounds to satisfy the constraints $C^{GC,NL}(n, M, d)$. In Table 3, the values in parenthesis with superscripts 'a' and 'k' indicate the Altruistic and KMVO's lower bounds, respectively. In contrast, the bold black values are outperformed the bounds values of the MFOL algorithm. A plethora of lower bounds delivered by the MFOL algorithm are better than the existing algorithm. For instance, at $n = 10$ and $d = 5$, the size of our DNA coding set is 25% better than the KMVO algorithm. In all sequences with $d = 4$, new DNA codes are 37% higher than the Altruistic algorithm and 30% better than the KMVO algorithm. Overall, MFOL enhanced 30% and 17% DNA coding sets in the given boundary compared to the Altruistic and the KMVO algorithm, respectively. These significant improvements are due to the consideration of the OBL mutation strategy with MFO, which empowers the exploration and optimization ability of the MFOL algorithm. In addition, Table 4 presents the DNA coding sets satisfying the $C^{GC,NL}(n, M, d)$ constraints when $n = 10$ and $d = 7$.

Furthermore, these improvements of lower bounds for the given sequence length are directly advantageous to the improvements of DNA coding rates. The coding rate (R) defines as; $R = \log_4 K/n$, where K is the number of DNA coding sets, and n is the number of sequence lengths [13]. The analysis of Table 5 indicates that the improved MFOL algorithm attained the same coding rate with shorter sequence lengths in 89% coding sets. For instance, the Altruistic algorithm achieved the coding rate $R = \log_4 86/8 = 0.4016$ when $n = 8$ and $d=4$. In contrast, the proposed algorithm reported a 0.4010 coding rate when $n = 7$

Table 3. Comparison of lower bounds of MFOL algorithm with Altruistic and KMVO algorithms for $C^{GC,NL}(n, M, d)$.

n	d = 3	d = 4	d = 5	d = 6	d = 7	d = 8	d = 9
4	11 (11 ^a)						
5	24 (20 ^k)	8 (7 ^a)					
6	58 (44 ^a)	26 (16 ^a)	7 (6 ^a)				
7	148 (127 ^k)	49 (36 ^a)	19 (11 ^a)	6 (4 ^a)			
8	328 (289 ^a)	114 (94 ^k)	35 (32 ^k)	11 (9 ^a)	6 (4 ^k)		
9	906 (680 ^k)	281 (202 ^k)	83 (65 ^k)	30 (23 ^k)	9 (8 ^a)	4 (4 ^a)	
10	2254 (2081 ^k)	721 (547 ^k)	189 (151 ^k)	79 (54 ^k)	17 (7 ^a)	5 (5 ^a)	5 (4 ^a)

Table 4. DNA storage coding sets when n = 10 and d = 7.

GACACTATAG	CTATACAGTG	AGCACATGAC	TATGCTACAT
ATCACACAGT	ATACAGCGAT	GAGTATACAT	ATAGCACATC
CTACTGACTA	CAGCATGATC	ATAGCAGATG	TACACGATAC
GTCACGTA CT	CAGTAGAGCA	GACGATGCTG	ATGCATCGAT
TCTAGCATCA			

and d = 4. Similarly, the KMVO algorithm attained a 0.5227 coding rate when n = 9 and d = 3. In comparison, the MFOL algorithm has a 0.5223 coding rate when n = 8 and d = 3. Thus, it analytically proved that shorter sequences can also accomplish the same DNA storage performance as longer sequences. It indicates that shorter sequences are less expensive and easier to synthesize with more stable conditions, which shows the improved lower bounds effectiveness for the further deployment of the DNA data storage system.

4.5 Temperature Variance of DNA Codes

The validity of DNA coding constraints is empirically computed by the temperature variance of DNA coding sets. In the coding of DNA storage, the melting temperature (T_m) is a certain temperature when half of the double-strand DNAs convert into single-strand DNAs during the denaturation process [20]. T_m depends on GC content, which affects the reaction rates of DNA molecules: the higher GC content presents, the higher T_m. In PCR amplification, sufficiently lower T_m can be more effective in binding the forward and reverse primers. Thus, both primers must be having similar T_m that can avoid the non-specific hybridization possibility. The non-specific hybridization is associated with oligonucleotides structure and its thermodynamic properties. Therefore, each DNA sequence should be with the same T_m to construct the DNA coding sets. The temperature variances are utilized to distinguish the sequence quality; the smaller the temperature variance, the more stable the T_m of the DNA coding set [21].

As the primary focus of this study was to construct the DNA codes with shorter sequences, an empirical thermodynamic test is conducted to validate the DNA sequences. For the temperature variance, the empirical values of primer

Table 5. Comparison of $C^{GC,NL}$ for Tm variance of DNA codes with $5 < n < 10$ and $2 < d < 9$.

n / d	Constraints	d=3	d=4	d=5	d=6	d=7	d=8
6	$C^{GC,NL}$	3.6311	4.2734	4.0246			
7	$C^{GC,NL}$	5.3691	4.1368	3.6814	4.7168		
8	$C^{GC,NL}$	3.9812	3.2451	3.1931	3.0161	4.364	
9	$C^{GC,NL}$	4.6841	5.3017	5.8054	2.8972	3.9218	2.7204

concentration are set at 200nM while the salt concentration is set at 50nM. For example, based on these concentrations, a primer (TATGTAGTAC) with sequence length 10 delivers the 30% GC-content, and nucleotides degeneration is allowed at $T_m = 26^\circ\text{C}$. The coding sets with the proposed MFOL algorithm are analyzed with existing constraints for its correlated Tm values. Table 12 compared the Tm variances with $C^{GC,NL}$ constraints for $5 < n < 10$ and $2 < d < 9$ lower bounds. The results present the significantly lowest Tm variance for the $C^{GC,NL}$. This analysis signifies the practical implication and necessity of the MFOL algorithm with the OBL strategy for DNA coding sets. These smaller Tm variances of DNA coding set advantageous the more stable PCR reaction due to reduction of non-specific hybridization.

5 Conclusion & Future work

This paper proposed a novel MFOL algorithm based on MFO that is synergized by the OBL strategy to construct the DNA coding sets. In experiments, the MFOL's exploration and exploitation capabilities are compared with 4 different state-of-the-art optimization algorithms. Based on MFOL's competent results (Tables 1 & 2 and Fig. 5), MFOL is applied in practical problems to generate DNA codewords with GC-content, Hamming distance, No-runlength constraints. It improved 30% and 17% lower bounds DNA coding sets compared to the Altruistic and the KMVO algorithms, respectively (Table 3). Meanwhile, the temperature variance with $C^{GC,NL}$ constraints for the given lower bounds also reported practical implications and the necessity of the MFOL algorithm for DNA data storage (Table 5). It is concluded that improved lower bounds can avoid further non-specific hybridization, and the shorter sequences can reduce more errors during the DNA synthesis and sequencing.

In the future, the DNA sequences generated by the MFOL algorithm will be assessed by the combinatory constraints of GC and RC. As the OBL strategy significantly improved the proposed algorithm capabilities, this work can be further extended by testing Levy flight or Cauchy mutation strategies with more benchmark functions (F1-F19). The effective efficiency of this stochastic algorithm will support the generation of the DNA codewords for a DNA-based data storage system.

References

1. David Reinsel, J.G., John Rydning, Data Age 2025: The Digitization of the World From Edge to Core, in An IDC White Paper. Nov. 2018, IDC.
2. Erlich, Y. and D. Zielinski, DNA Fountain enables a robust and efficient storage architecture. *Science*, 2017. 355(6328): p. 950-953.
3. Bornholt, J., et al., Toward a DNA-based archival storage system. *Ieee Micro*, 2017. 37(3): p. 98-104.
4. Church, G.M., Y. Gao, and S. Kosuri, Next-Generation Digital Information Storage in DNA. *Science*, 2012. 337(6102): p. 1628-1628.
5. Goldman, N., et al., Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature*, 2013. 494(7435): p. 77-80.
6. Li, M., et al., A self-contained and self-explanatory DNA storage system. *Scientific Reports*, 2021. 11(1): p. 18063.
7. Song, W., et al., Codes With Run-Length and GC-Content Constraints for DNA-Based Data Storage. *Ieee Communications Letters*, 2018. 22(10): p. 2004-2007.
8. Mirjalili, S., Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, 2015. 89: p. 228-249.
9. Emary, E., et al., Firefly Optimization Algorithm for Feature Selection, in Proceedings of the 7th Balkan Conference on Informatics Conference. 2015, Association for Computing Machinery: Craiova, Romania. p. Article 26.
10. Mirjalili, S., S.M. Mirjalili, and A. Lewis, Grey Wolf Optimizer. *Advances in Engineering Software*, 2014. 69: p. 46-61.
11. Heidari, A.A., et al., Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, 2019. 97: p. 849-872.
12. Mirjalili, S., S.M. Mirjalili, and A. Hatamlou, Multi-Verse Optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications*, 2016. 27(2): p. 495-513.
13. Cao, B., et al., K-Means Multi-Verse Optimizer (KMVO) Algorithm to Construct DNA Storage Codes. *Ieee Access*, 2020. 8: p. 29547-29556.
14. Limbachiya, D., M.K. Gupta, and V. Aggarwal, Family of Constrained Codes for Archival DNA Data Storage. *Ieee Comm. Letters*, 2018. 22(10): p. 1972-1975.
15. Abolunio, N., D.H. Smith, and S. Perkins, Linear and nonlinear constructions of DNA codes with Hamming distance d , constant GC-content and a reverse-complement constraint. *Discrete Mathematics*, 2012. 312(5): p. 1062-1075.
16. Abualigah, L., et al., The Arithmetic Optimization Algorithm. *Computer Methods in Applied Mechanics and Engineering*, 2021. 376.
17. Tizhoosh, H.R. Opposition-Based Learning: A New Scheme for Machine Intelligence. in International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06). 2005.
18. Rasool, A., et al., GAWA—A Feature Selection Method for Hybrid Sentiment Classification. *IEEE Access*, 2020. 8: p. 191850-191861.
19. Kim, D.H. and Y.C. Kim, Wilcoxon signed rank test using ranked-set sample. *Korean Journal of Computational & Applied Mathematics*, 1996. 3(2): p. 235-243.
20. Chee, Y.M. and S. Ling, Improved Lower Bounds for Constant GC-Content DNA Codes. *IEEE Transactions on Information Theory*, 2008. 54: p. 391-394.
21. Sager, J. and D. Stefanovic. Designing Nucleotide Sequences for Computation: A Survey of Constraints. in DNA Computing. 2006. Berlin, Heidelberg: Springer Berlin Heidelberg.