Received January 9, 2020, accepted January 26, 2020, date of publication January 31, 2020, date of current version February 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2970838

K-Means Multi-Verse Optimizer (KMVO) Algorithm to Construct DNA Storage Codes

BEN CAO, SUE ZHAO, XUE LI, AND BIN WANG
Key Laboratory of Advanced Design and Intelligent Computing, Ministry of Education, School of Software, Dalian University, Dalian 116622, China

Corresponding author: Bin Wang (wangbin@dlu.edu.cn)

This work was supported in part by the National Key R&D Pro-gram of China under Grant 2018YFC0910500, in part by the National Natural Science Foundation of China under Grant 61425002, Grant 61751203, Grant 61772100, Grant 61972266, Grant 61802040, Grant 61672121, and Grant 61572093, in part by the Program for Changjiang Scholars and Innovative Research Team in University under Grant IRT_15R07, in part by the Program for Liaoning Innovative Research Team in University under Grant LT2017012, in part by the Natural Science Foundation of Liaoning Province under Grant 20180551241 and Grant 2019-ZD-0567, in part by the High-Level Talent Innovation Support Program of Dalian City under Grant 2017RQ060 and Grant 2018RQ75, in part by the Dalian Outstanding Young Science and Technology Talent Support Program under Grant 2017RJ08, and in part by the Scientific Research Fund of Liaoning Provincial Education Department under Grant JYT19051.

ABSTRACT In an era of information explosion, dealing with massive data has become a problem. Since DeoxyriboNucleic Acid (DNA) is a high-density storage medium with long storage endurance, a DNA based storage system is a viable solution. The first consideration of a DNA storage system is the DNA codes, which can avoid non-specific hybridization of DNA strands in the hybridization reaction process by using related constraints, such as Hamming distance constraints, GC-content constraints, and no-runlength constraints. A K-means Multi-Verse Optimizer (KMVO) algorithm is proposed to construct a better code boundary than the previous Multi-Verse Optimizer (MVO) algorithm that satisfies the above constraints. Our results can store information more efficiently over a given length, increasing storage utilization.

INDEX TERMS DNA code design, DNA storage, k-means, MVO algorithm.

I. INTRODUCTION

Methods of data storage trace back to ancient times, when people used a rope to record important information. In modern times, people use CDs, floppy disks, and hard disks to store information. However, with the development of industry and information technology, the exponential growth of data has become a challenge to data-storage technology [1]. If people want to store and utilize more information, they need denser, less costly storage media [2]. Electronic science and technology have also led to the growth of e-waste, so we need to find other storage media [3]. As a high-density storage medium, DNA has high storage capacity and long-term stability. With four nucleotides (*nt*), including adenine (A), thymine (T), cytosine (C), and guanine (G) [4], DNA as storage medium capacity is twice binary, and the theoretical information density about 1,018 B/mm^3. Moreover, DNA data can be stored for many years under adaptive conditions. Nevertheless, the cost of reading and writing DNA data remains high. But, with the recent rapid development of DNA synthesis and sequencing methods, DNA storage has potential as a competitive storage solution.

Bornholt *et al.* [5] used wet experiments to demonstrate that the polymerase chain reaction can be used to randomly access a DNA storage pool while also indicating the long-term effectiveness of DNA as an archival storage medium. Nguyen *et al.* [6] presented a report on the long-term stability and integrity of plasmid-based DNA data storage. They used a Perl script to encode a 2,046-word DNA sequence and synthesized the encoded DNA sequence to store the information. The plasmid DNA was placed under accelerated aging conditions (AAC) and showed no differences up to 65 °C for 20 days. Finally, the source data were retrieved by sequencing and decoded, and the text was read with 100% accuracy, proving the long-term stability and integrity of Plasmid-Based DNA data storage technique. Tomek *et al.* [7] used chemical methods to extract unique files from a complex DNA database that mimicked 5 terabytes of data, and designed and tested a nested file address system. The theoretical capacity of a DNA storage system has increased by five orders of magnitude than previous work. Chen *et al.* [8] increased the density of long-term DNA storage, using silica

The associate editor coordinating the review of this manuscript and approving it for publication was Quan Zou.

spheres to reach 10 times the current state-of-the-art. A layer-by-layer (*LbL*) design was used to bond magnetic nanoparticles through alternating layers, while a protective silicon layer was grown on top to protect the DNA from external sources of damage. Accelerated aging showed that the degradation rate was significantly reduced compared with the unprotected DNA of a control group. Takahashi *et al.* [9] realized the 5-byte automatic write, save and read functions through the modular design of DNA storage, which can be extended by new technologies. Wang *et al.* [10] monitored the long-term storage of DNA by ddPCR. Heckel *et al.* [11] found that intramolecular errors were mainly due to synthesis and sequencing, qualitative and quantitative analysis of DNA data storage channels helps guide the design of future DNA data storage systems. Other interesting ways of storing DNA have been proposed, using bio-binding codes [12] and electronic bio-mixing systems [13].

DNA coding focuses on quality and quantity. By building higher-quality code and larger code sets, we can solve larger problems and get more reliable results. Regrettably, these two indicators are contradictory, and the number of codes will decrease as their quality increases. The primary objective in DNA storage is to avoid non-specific hybridization of a set of DNA codes. Second, it is necessary to construct a sufficiently large code set under the condition that the Combination constraint is satisfied, so our focus is on the number of codes. The problem of constructing DNA storage codes translates to one of constructing the largest set of coded sequences.

The Microvenus project initiated by Joe Davis aims to store non-biological data such as images in DNA and encode them based on molecular sizes *CTAG* (*C-1, T-2, A-3, G-4*) [14], e.g., 10101→CCCC, 100101→CTCCT. However, due to a flaw in decoding, *C* can decode to 0 or 1, resulting in many errors [15]. This pioneering DNA-storage method has not been widely used because inconsistencies before and after decoding lead to errors [16]. Garzon and Deaton provided a complete definition of code problems in DNA computing [17]. (1) The reaction product satisfies the constraints. (2) No errors occur during the biochemical reaction. Ross *et al.* [18] proposes that the reaction product not only requires high stability but also can be successfully decoded into the solution of the original problem. A DNA-based storage system was designed to support random access and arbitrary locations for rewriting data blocks to overcome the shortcomings of current read-only systems [19]. In DNA storage, information is stored as oligonucleotides. It has been confirmed that an operation with too high or too low GC content is prone to errors [17]. Therefore, scientists believe that DNA sequences are robust to errors in data storage procedures when GC content is locally stable at 50%. Hong *et al.* [20] introduced algebraic number theory to DNA codes de-sign, and obtained a set of larger DNA codes set that satisfy the constraints of GC content and the code set content. Gabrys *et al.* [21] introduced an asymmetric error-correction code to correct errors based on DNA systems and transmission systems. Song *et al.* [22] used two binary bits to map directly to one nucleotide. Although the code rate is close to the theoretical channel capacity, processing after iteration may cause serious transmission errors during decoding. Immink *et al.* [23] designed a constraint code with a capacity close to that to avoid this situation, and the long homopolymer that appeared was replaced by the sequence method to run, but a GC-content constraint was not considered. Bornholt *et al.* [5] proposed a new method to link short codes that only satisfy homopolymer operating constraints with long DNA sequences. However, the GC content and complexity of DNA short chains in long sequences were not guaranteed. Yazdi *et al.* [24] deduced the size of the WMU and the boundaries of the various constrained WMU codes to avoid primer dimers.

We propose a K-means Multi-Verse Optimizer (KMVO) algorithm to improve the situation of code constraints in DNA storage. DNA codes can reduce non-specific hybridization between different codes and have the three main constraints of Hamming distance, GC content, and no-runlength. We use the KMVO algorithm to construct the codes in the DNA store. Our improved algorithm is inspired by the recent wormhole theory [25] and theory of the livable planetary agglomeration belt. Based on the latter, the idea of clustering is introduced to Multi-Verse Optimizer (MVO) to accelerate the convergence rate. Wormhole crossing makes the MVO algorithm jump out of local optima and expand the search range.

The remainder of this paper is organized as follows. Section 2 describes the constraints of the code set in DNA storage. Section 3 introduces the mechanism and model of the MVO algorithm and our improvement based on k-means. K-means is a clustering algorithm with simple formula and good classification effect. Section 4 includes the results and analysis. Section 5 concludes the article with a general outlook.

II. DNA CODE CONSTRAINTS

The purpose of the code set design in DNA storage is to construct a collection of DNA strands of a given length *n*. These generated code words form a code set that makes more efficient use of DNA bases. We wish to make the set of a given length as large as possible, make the DNA more stable, and reduce the error in the equation.

A. HAMMING DISTANCE CONSTRAINT

For any pair of DNA sequences x, y in the set, the Hamming distance constraint is denoted as H(x,y) $\geq$ d, where H(x,y) denotes the number of positions at which the corresponding symbols are different in between x and y [26]. The Hamming distance is calculated as

$$H(x,y)=\sum_{i=1}^{n}h(x_i,y_i),\ h(x_i,y_i)=\begin{cases}0, x_i=y_i\\1, x_i\neq y_i\end{cases}\tag{1}$$

The Hamming distance is used to describe the magnitude of the similarity of the two sequences, and the smaller the value, the higher the similarity. This means that the fewer the number of different bases between the two DNA codes, the greater the number of identical bases; hence, there is a greater probability of non-specific hybridization between DNA sequences.

B. GC-CONTENT CONSTRAINT

The GC-content constraint is the ratio of the total number of bases G and C in a DNA strand. Generally, GC-content is about 50% and is not prone to error and stability. Here, we set GC-content to 50% [27].

The GC-content for a sequence of length x is denoted as *GC(x)*. The GC-content constraint specifies that $GC(x) = \lfloor n/2 \rfloor$, we use the following formula to calculate the GC-content:

$$GC(x) = \frac{|G| + |C|}{|x|}. \tag{2}$$

where $|G|$ and |C| respectively represent the number of G and C in the code, and $|x|$ is the length of sequence x.

C. NO-RUNLENGTH CONSTRAINT

The DNA code should not include duplicate bases. Running the same nucleotides for a long time can lead to errors in DNA codes [28]. For example, in *ATTTAC*, T is repetitive, so in synthesis and sequencing, it is easy to read a long T as a short T, increasing the loss rate of DNA information and reducing the read/write coverage. There is a sequence A (a_1, a_2, $a_3 \ldots a_n$) of length n:

$$\mathrm{A_i} \neq \mathrm{A_{i+1}} \quad \mathrm{i} \in [1,\mathrm{n}\text{-}1] \tag{3}$$

We define $\mathrm{A}^{GC,NL}(n, d, w)$ as the largest set of DNA codes that satisfy the GC-content and no-runlength constraints for given parameters *n, d*.

III. ALGORITHM DESCRIPTION

A. MVO ALGORITHM

The Belgian astronomer and cosmologist Georges Lemaitre proposed the big bang hypothesis in 1927. This theory suggests that the universe originated from a huge explosion, *i.e.*, everything came from a big bang [29]. In a multiverse, multiple universes interact, such as by attracting each other and colliding. The multiverse theory has the key concepts of black holes, white holes, and wormholes, which inspired the MVO algorithm [30]. The universe is always expanding, so a universe has an expansion rate (eternal expansion). This has a big impact on the properties of matter, such as the adaptability of life to planets and asteroids, and the laws of physics (*e.g.*, the gravitational acceleration of Mars is 4/9 that of Earth). The cyclic multi-universe model [31] states that multiple universes can exchange matter through black/white holes to achieve a stable state. The concept underlying MVO [30] is that matter moves from low-complexity universes to highly adaptive universes, increasing the average fitness value of all solutions.

To better establish a model for multiverse theory, we first sort the universes according to the standardized inflation rate and then use roulette during each iteration to select a universe with a white hole. The search space is explored by the black/white hole mechanism. The higher a universe's expansion rate, the more likely it is to create white holes and to transfer objects [30]. We believe that wormholes can transport substances at will without considering the expansion rate. We assume that wormholes are frequently established between the current universe and the best universe. The expression of the wormhole is

$$xij = \begin{cases} \begin{cases} Xj + TDR \times ((ubj - lbj) \times r4 + lbj), \, r3 < 0.5 \; r2 < WEP \\ Xj - TDR \times ((ubj - lbj) \times r4 + lbj), \, r3 \geq 0.5 \end{cases} \\ xijr2 \geq WEP, \end{cases} \tag{4}$$

where X_j denotes the *jth* substance of the best universe currently created, the boundaries of the *jth* material are ub_j and lb_j, and r_2, r_3, and r_4 are random numbers in the interval [0, 1].

In the MVO algorithm [30], a series of random universes are first initialized. Matter in a universe can be transmitted in two ways at each iteration. First, each universe tends to produce white holes and optimal universe material exchange. Second, through the black/white tunnel, a high-expansion universe transmits matter to a low-expansion universe. In this paper, the relevant parameters (*TDR, WEP, p*) are consistent with the literature [30]:

$$WEP = \min + (\max - \min) \times \frac{l}{L} \tag{5}$$

$$TDR = 1 - \frac{l^{1/p}}{L^{1/p}} \tag{6}$$

B. KMVO ALGORITHM

In the MVO algorithm, an individual's update mainly depends on the size of the expansion rate, and random updates occur based on the current optimal global and parameter *WEP*. Since the optimal value in the early stages of the algorithm is often too far from the true value, this strategy will increase the probability that the algorithm will fall into a local optimum and may cause the convergence speed to decline.

To solve the above problems, we introduce the worm-hole theory and the theory of the livable planetary agglomeration belt. Based on the theory of the livable planetary agglomeration belt [32], we assume there may be a universe-gathering belt, whose different universes have similar characteristics, in the multiverse. From this conjecture, we introduce the k-means clustering algorithm to MVO and cluster the initial universe, hoping to improve the performance of MVO. The division of objects is practical in many fields such as statistics, biology, and computational science. Dividing several objects into k categories is the basic idea of partitioning [33]. As number of objects increases, it may not be practical to list all of number. Nevertheless, the small number of initial universes in our algorithm is suitable for this simple and efficient calculation method. It is valuable to provide division within a reasonable calculation time. The

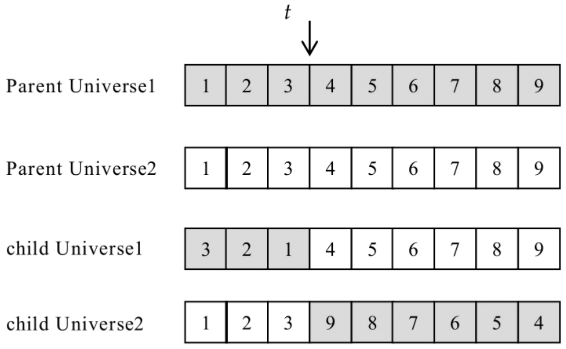


FIGURE 1. Wormhole cross.

TABLE 1. Meanings of parameters.

Parameter	value
min	0.2
max	1
p	6

k-means clustering algorithm was proposed in the 1950s, it is still popular [34-37], and researchers are interested in improving it. The algorithm divides bidirectional bimodal data into k classes (n_1, n_2, $n_3 \ldots n_k$), where n_k is the set of n_k objects in the k category, and k is the set category to be divided. This is useful for solving problems and classifying them before optimization. The advantage of k-means is its low complexity and its simple, effective formula [38]. We use k-means to overcome MVO's slow convergence speed, speed up the optimization, and reduce the number of iterations. At the same time, we improved the wormhole cross in the MVO algorithm according to the latest wormhole theory [25]. A wormhole can instantaneously transfer matter between any two universes. Then wormholes may not only be created between the universe and the best universe, but other universes may also have wormholes. As shown in Fig.1, creating a black hole at the t position causes *Universe1* and *Universe2* to exchange matter under the action of a black hole. We call this a wormhole cross.

In the KMVO algorithm, each time the universe position is updated, the universe is clustered into the optimal and worst class, these are wormhole crossed, and the result is entered into the next iteration together with the optimal class. The wormhole cross can not only enrich the diversity of the universe but also jump out of a local optimum. The pseudo-code and flowchart Fig.2 are as follows.

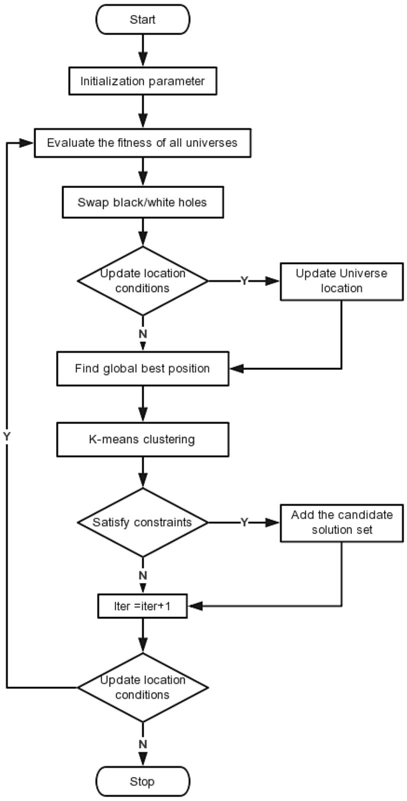

Start
Initialization parameter
Evaluate the fitness of all universes
Swap black/white holes
Update location conditions
Y
Update Universe location
N
Find global best position
K-means clustering
Satisfy constraints
Y
Add the candidate solution set
N
Iter =iter+1
Update location conditions
Y
N
Stop

FIGURE 2. KMVO algorithm flowchart.

TABLE 2. Unimodal benchmark functions.

Function	Dim	Range	$F_{\min}$
$F_1(x)=\sum_{i=1}^{n} X_i^2$	50	[-100,100]	0
$F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	50	[-10,10]	0
$F_3(x)=\sum_{i=1}^{n}(\sum_{j-1}^{i} x_j)^2$	50	[-100,100]	0
$F_4(x)=\max_i\{\lvert x_i\rvert,1\le i\le n\}$	50	[-100,100]	0
$F_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	50	[-30,30]	0
$F_6(x)=\sum_{i=1}^{n}([x_i+0.5])^2$	50	[-100,100]	0
$F_7(x)=\sum_{i=1}^{n} ix_i^4+random[0,1)$	50	[-1.28,1.28]	0

C. EXPERIMENT ENVIRONMENT

A simulation test was run on an Intel Core i7 3.6-GHz CPU with RAM 4 GB in MATLAB 2018a; the results are shown in Tables 3-8. Map four bases to numbers 0-3 when we construct a DNA code set (*T-0, C-1, G-2, A-3*) in KMVO program. Some key parameters and their meanings are shown in Table 1.

D. BENCHMARK FUNCTIONS

To test the performance of the improved algorithm, we used 13 benchmark functions [39]–[41] for test comparison. No algorithm can solve all problems, and the test function can't get the best results. One algorithm cannot achieve the best result on all test functions. We selected 13 test

TABLE 3. Multi-modal benchmark functions.

Function	Dim	Range	F_{min}
$F_8(x)=\sum_{i=1}^{n}-x_i\sin(\sqrt{\lvert x_i\rvert})$	50	[-500,500]	-418.9829*5
$F_9(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	50	[-5.12,5.12]	0
$F_{10}(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	50	[-32,32]	0
$F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}x_i^2\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	50	[-600,600]	0
$F_{12}(x)=\frac{\pi}{n}\left\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})+(y_n-1)^2]\right\}+\sum_{i=1}^{n}u(x_i,10,100,4)$ $y_i=1+\frac{x_i+1}{4}$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	50	[-50,50]	0
$F_{13}(x)=0.1\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{n}\begin{matrix}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+\\(x_n-1)^2[1+\sin^2(2\pi x_n)]\end{matrix}\right\}$	50	[-50,50]	0

functions including 7 high-dimensional unimodal functions and 6 high-dimensional multimodal functions. These 13 functions are representative of optimization problems, and testing with them can well explain the optimization performance of the algorithm. To improve the credibility of the test results, we placed constraints on the range of values of the test functions.

The 13 test functions were each run 30 times, and their average values and standard deviations were compared. Among them, PSO is a stochastic optimization algorithm that imitates group behavior, GA is the first representative evolutionary algorithm, and GSA is the best algorithm based on physics. The maximum number of iterations was set to 500. The results of MVO, GWO, GSA, PSO, and GA are from a previous work, namely Mirjalili's work [30]. Tables 2 and 3 list the test functions used. We also performed Wilcoxon's nonparametric rank sum detection and evaluated the results. Due to the randomness of the heuristic algorithm, the statistical run sum test result p is more convincing for the advantages of the algorithm. When $p>0.05$, the advantage of the KMVO algorithm is statistically significant in most cases. F1-F7 are high-dimensional unimodal functions with global optimality, so they are suitable for universal testing of the algorithm. F7-F13 are high-dimensional multi-modal functions with multiple local optimal solutions and one global optimal solution, and the number of local optimal solutions increases with the dimension. This adds difficulty to the solution, and it can better reflect the ability of an algorithm to search and to jump out of local optima.

IV. RESULTS

A. HIGH-DIMENSIONAL UNIMODAL FUNCTION

To test the performance of KMVO, it was compared with algorithms such as MVO and PSO. Tables 4 and 5 list the results of each algorithm running independently 30 times for F1-F7.

From Tables 4 and 5, both the average value and standard deviation of proposed algorithm are smaller than other algorithms for functions F2 and F5. Compared with the MVO algorithm, the mean for F2 is reduced by four times, and the variance by two orders of magnitude. The standard deviation of the test function for F4 is zero. The results of each run are stable at 30 runs, indicating the superiority of KMVO. For the test F3, our algorithm lags behind the previous MVO algorithm. This may be due to the large optimization interval of the F3 function, for which our algorithm did not converge well in the early stage.

B. HIGH-DIMENSIONAL MULTIMODAL FUNCTION

Tables 6 and 7 list the results of F7-F13 running 30 times independently. Compared with the unimodal function, the multimodal function has more local optimal solutions,

TABLE 4. Average result of unimodal benchmark functions.

F	KMVO	MVO	GWO	GSA	PSO	GA
	Ave	Ave	Ave	Ave	Ave	Ave
F1	8.8094	2.08583	2319.19	2983.667	3.552364	27,187.58
F2	**3.1771**	15.92479	14.43166	10.96518	8.716272	68.6618
F3	2209.7671	453.2002	7278.133	113,740.40	2380.963	48,530.91
F4	**1**	3.123005	13.09729	32.2563	21.5169	62.99326
F5	**603.1599**	1272.13	3,425,462	7582.498	1132.486	65,361,620
F6	9.3446	2.29495	5009.442	74,617.45	86.62074	49,574.10
F7	0.11137	0.051991	0.408082	21.16092	0.577434	18.72524

TABLE 5. SD result of unimodal benchmark functions.

F	KMVO	MVO	GWO	GSA	PSO	GA
	SD	SD	SD	SD	SD	SD
F1	1.6306	0.648651	1237.109	903.3827	2.853733	2745.82
F2	**0.72651**	44.7459	5.923015	10.54968	4.929157	6.062311
F3	786.9794	177.0973	2143.116	78,786.15	1183.351	8249.75
F4	**0**	1.582907	11.3469	6.226765	6.71628	2.535643
F5	**611.1058**	1479.477	3,304,309	7314.818	1357.967	29,714,021
F6	2.3629	0.630813	3028.875	8231.224	147.3067	8545.149
F7	**0.028805**	0.029606	0.119544	12.1566	0.318544	4.935256

TABLE 6. Average result of multi-modal benchmark functions.

F	KMVO	MVO[30]	GWO[30]	GSA[30]	PSO[30]	GA[30]
	Ave	Ave	Ave	Ave	Ave	Ave
F8	**-12,348.6192**	-11,720.20	-10,739.50	-4638.41	-6727.59	-10,698.60
F9	**49.9577**	118.046	89.13475	128.0103	99.83202	273.2519
F10	**2.9395**	4.074904	9.452571	1.654073	4.295044	18.59657
F11	**0.75229**	0.938733	22.51942	1021.705	624.3092	353.3655
F12	**1.898**	2.459953	3,200,008	741,596.90	13.38384	2.21E-08
F13	**1.35E-32**	0.222672	7,815,082	6,670,046	21.11298	4.49E-08

TABLE 7. SD result of multi-modal benchmark functions.

F	KMVO	MVO[30]	GWO[30]	GSA[30]	PSO[30]	GA[30]
	SD	SD	SD	SD	SD	SD
F8	**724.6721**	937.1975	1162.793	805.0488	1352.882	602.3045
F9	**0.017879**	39.34364	37.95765	26.90054	24.62872	29.55218
F10	**0.57575**	5.501546	3.467608	1.583499	1.308386	0.351737
F11	**0.049873**	0.059535	26.68168	82.95486	105.3874	77.26729
F12	**0.61601**	0.791886	6,746,208	624,375.50	8.969122	1.10E-08
F13	**5.57E-48**	0.086407	16,475,640	5,719,826	12.83179	2.26E-08

which presents certain obstacles to an algorithm's solving function. Due to the complexity of the high-dimensional multimodal function, accurate results are more important than fast convergence. The number of local optimal solution of the high-dimensional multimodal function also increases with the dimension.

KMVO Pseudo-Code

```
for each universe indexed by i
    Update WEP and TDR
    Black_hole_index = i;
    for each object indexed by j
        r1 = random([0,1]);
        if r1<NI(U i)
            White_hole_index =
            RouletteWheelSelection(-NI);
            U(Black_hole_index,j) =
            SU(White_hole_index,j);
        end if
        $r_2$= random([0,1]);
        if $r_2$<Wormhole_existence_probability
            $r_3$= random([0,1]);
            $r_4$= random([0,1]);
            if $r_3$<0.5
                U(i,j) = Best_universe(j)+Travelling_
                distance_rate* ((ub(j)- lb(j)) * $r_4$
                + lb(j));
            else
                U(i,j) = Best_universe(j)+Travelling_
                distance_rate* ((ub(j)- lb(j)) * $r_4$
                + lb(j));
            end if
        end if
    end for
    If Time<MaxTime/2
        Buniverse = clustering(Universes, Best_universe);
        Buniverse = exchange(BUniverses,1);
    end if
end for
```

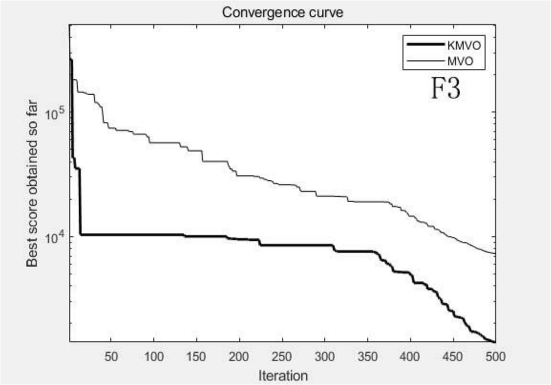


FIGURE 3. The convergence curve is compared at F3.

From Tables 6 and 7, the average value and the standard deviation of proposed algorithm are significantly lower than MVO [30]. The mean and variance of F13 are close to zero. This is a good result of using k-means clustering in the early stage of the algorithm. For F11, the optimization function has a large optimization range and the previous MVO algorithm has close to 0.001, so our algorithm does not greatly improve the result of the function. The significance of comparing the mean and standard deviation with other algorithms is that averaging the results of multiple runs can reduce the impact of accidental results, and the standard deviation indicates whether the algorithm is stable in 30 runs. A small standard deviation means that the differences between runs are small, and the smaller the difference, the more stable the result. In order to observe the convergence behavior of KMVO algorithm, the convergence curves of two different types of benchmark functions are given respectively. As shown in Fig.3 and Fig. 4, KMVO algorithm has achieved good results compared with MVO algorithm in both exploration and exploitation stages. Especially in F3, when the convergence curve tends to be horizontal, that is, no longer converge, KMVO algorithm continues to search for the optimal solution by jumping out of the local optimal through k-means clustering.

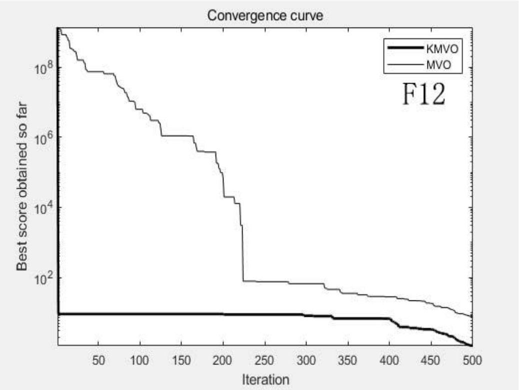


FIGURE 4. The convergence curve is compared at F12.

C. WILCOXON'S NON-PARAMETRIC RANK SUM

We use the rank-sum test to evaluate the quality of pro-posed algorithm, and it does not depend on the specific form of the overall distribution. The rank sum test does not depend on the specific form of the overall distribution. It is practical in that it can be applied without regard to the distribution of the object being studied, or to whether the distribution is known [45].

To avoid defects, Wilcoxon proposed an improved method, the Wilcoxon rank sum test [46]. This method considers the direction and size of the difference, which is more effective than the symbol test. A similar method can be used to check for differences in the distribution positions of the test data. The rank sum test ranks all observations in ascending order, and the number of each observation in order is called its rank.

Calculate the rank sum for any two of the 30 rounds. This method compares the rank of each pair, which improves its testing efficiency. We performed Wilcoxon's nonparametric rank sum test on 13 test functions, and the ideal result was $p>0.05$, for which we believe that p rejects the null hypothesis and proves that the results are very competitive. As can be seen in Table 8, our results rejected the hypothesis in most cases.

TABLE 8. P values of Wilcoxon rank sum test over 30 runs.

F	KMVO	MVO[30]	GWO[30]	GSA[30]	PSO[30]
F1	0.34817	N/A	0.002827	0.000183	0.185877
F2	0.51033	0.009108	0.053903	0.909722	N/A
F3	0.87769	N/A	0.000183	0.000183	0.000183
F4	N/A	N/A	0.140465	0.000183	0.000183
F5	N/A	0.677585	0.10411	0.005795	N/A
F6	0.58492	N/A	0.000183	0.000182	0.007284
F7	0.62052	N/A	0.000183	0.000183	0.000183
F8	0.60117	N/A	0.053903	0.000183	0.000183
F9	0.26379	0.121225	N/A	0.002827	0.185877
F10	0.35904	0.121225	0.001315	N/A	0.002827
F11	0.59211	N/A	0.005795	0.000183	0.000183
F12	0.48449	N/A	0.025748	0.000183	0.00033
F13	N/A	N/A	0.075662	0.000183	0.000183

TABLE 9. Lower bounds for $A^{GC,NL}(n, d, w)$.

n\d	3	4	5	6	7	8	9	10
4	11^a $\mathbf{12^k}$							
5	17^a $\mathbf{20^k}$	7^a $\mathbf{8^k}$						
6	44^a $\mathbf{56^k}$	16^a $\mathbf{23^k}$	6^a $\mathbf{8^k}$					
7	110^a $\mathbf{127^k}$	36^a $\mathbf{45^k}$	11^a $\mathbf{17^k}$	4^a $\mathbf{6^k}$				
8	289^a $\mathbf{319^k}$	86^a $\mathbf{94^k}$	29^a $\mathbf{32^k}$	9^a $\mathbf{13^k}$	4^a $\mathbf{5^k}$	4^a 4^k		
9	662^a $\mathbf{680^k}$	199^a $\mathbf{202^k}$	59^a $\mathbf{65^k}$	15^a $\mathbf{23^k}$	8^a $\mathbf{10^k}$	4^a $\mathbf{5^k}$	4^a 4^k	
10	1810^a $\mathbf{2081^k}$	525^a $\mathbf{547^k}$	141^a $\mathbf{151^k}$	43^a $\mathbf{54^k}$	7^a $\mathbf{19^k}$	5^a $\mathbf{9^k}$	4^a 4^k	

TABLE 10. Meanings of superscripts.

Superscript	Meaning
a	Altruistic algorithm
k	KMVO algorithm

D. BOUNDS ON DNA STORAGE CODES

We define $A^{GC,NL}(n, d, w)$ as the set of DNA sequences with length n, Hamming distance d, satisfying the Hamming distance, GC-content, and no-runlength constraints. In Table 9,$A^{GC,NL}(n, d, w)$ is a lower bound satisfying the constraint $4 \le n \le 10$, $3 \le d \le n$. Bolding indicates a part of our algorithm that is better than altruistic algorithm [28]. The values are shown in Table 9, where the meanings of the superscripts are shown in Table 10. In addition, to make the results more convincing, we show the DNA coding set satisfying combination constraints at $n = 9$ and $d = 6$ in Table 11.

TABLE 11. DNA storage codes set in $n = 9$, $d = 6$.

TCTGTCGTA	*TACGAGTAC*
TCGCATACA	*TAGTCAGTC*
TGCACTGAT	*CTATCTCAC*
TGACTCTAG	*CTACGAGTA*
TATACGACG	*CATGATGCT*
CAGATGCTA	*CACTGTATG*
GTGTACTGT	*GCATAGATC*
GTAGTGTCA	*GATCTACAC*
ATCGTAGAG	*ATCTCGAGA*
AGTATCAGC	*ACGAGATAC*
AGATACGCA	*AGTCATCTG*
AGAGCATGT	

After comparing the size of DNA codes set with previous work, namely Limbachiya [28], most are better than those in the previous paper. As shown in Table 9, when $n = 10$ and $d = 7$, and the size of our DNA coding set is 1.5 times that of the previous DNA coding set. The significant improvement of DNA codes may be attributed to the strong exploration and development ability of MVO, and the k-means clustering algorithm improves the iterative optimization ability of the MVO algorithm. For the same results as before in the table 9, when d is close to n, KMVO cannot find more effective solutions because the previous results are close to the upper bound. Finding a larger code set for a given length cannot only decrease the code length but can increase the code rate, which is defined as $R = \log_4 M/n$ [47], where n is the length of the DNA code, and m is the number of DNA code sets. When $n = 8$, $d = 3$, the result of Limbachiya [28] is $R = \log_4 289/8 \approx 0.51$. In our method, when $n = 7$, $d = 3$, $R = \log_4 129/7 \approx 0.51$. The code rate reaches the same level under these two conditions. Therefore, shorter code lengths can achieve the same performance, and in our approach, increasing the size of the code set can improve storage performance.

V. CONCLUSION

We have proposed an improved MVO algorithm to construct DNA storage codes. The algorithm is inspired by the theory of multiverses, to which we have added planetary clusters and new wormhole ideas. The proposed algorithm was compared with MVO, GA, PSO, and other algorithms based on 13 test functions, and its statistical results (mean, standard deviation) were significantly improved. The rank sum test was also used to evaluate the statistical significance of the algorithm. Due to the randomness of the metaheuristic algorithm, it is necessary to reject the test of the null hypothesis. Through the comparison of test functions, our proposed algorithm has general applicability to other problems. DNA storage encoding is a sub-problem in DNA storage, which our algorithm can be used to solve. The combination constraints of DNA codes can effectively limit non-specific hybridization in a reaction. For this reason, we constructed a larger DNA code set than previous work under the combinatorial constraints. Simulation experiments show that in many cases our algorithm can construct a larger DNA code set than altruistic algorithm and avoid more non-specific hybridization. This further illustrates the superiority of our algorithm. We introduced the concept of a code rate and further compared the code sets. Our method can achieve the same code rate with a shorter sequence length, so it is efficient and more competitive than altruistic algorithm in DNA storage. In other aspects, DNA coding image encryption [48] and some neural-like computing models, see e.g. neural networks [49], [50] and parallel computing models [51], [52] can be considered to design reliable DNA code set in DNA storage.

In the future, we plan to continue to improve the MVO algorithm. In our algorithm operation, we can clearly see from the test functions F9 and F4 that the optimization process of the algorithm does not last until the end of the iteration. We will also continue to construct larger collections of DNA codes to store information more efficiently.

REFERENCES

[1] A. Siddiqa, A. Karim, and A. Gani, "Big data storage technologies: A survey," *Frontiers Inf. Technol. Electron. Eng.*, vol. 18, no. 8, pp. 1040–1070, 2017, doi: 10.1631/fitee.1500441.

[2] Y. Erlich and D. Zielinski, "DNA fountain enables a robust and efficient storage architecture," *Science*, vol. 355, no. 6328, pp. 950–954, Mar. 2017, doi: 10.1126/science.aaj2038.

[3] D. N. Perkins, M.-N. Brune Drisse, T. Nxele, and P. D. Sly, "E-waste: A global hazard," *Ann. Global Health*, vol. 80, no. 4, pp. 286–295, Nov. 2014.

[4] H. M. Kiah, G. J. Puleo, and O. Milenkovic, "Codes for DNA sequence profiles," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3125–3146, Jun. 2016, doi: 10.1109/tit.2016.2555321.

[5] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, "Toward a DNA-based archival storage system," *IEEE Micro*, vol. 37, no. 3, pp. 98–104, 2017, doi: 10.1109/mm.2017.70.

[6] H. Nguyen, J. Park, S. Park, C.-S. Lee, S. Hwang, Y.-B. Shin, T. Ha, and M. Kim, "Long-term stability and integrity of plasmid-based DNA data storage," *Polymers*, vol. 10, no. 1, p. 28, Jan. 2018.

[7] K. J. Tomek, K. Volkel, A. Simpson, A. G. Hass, E. W. Indermaur, J. M. Tuck, and A. J. Keung, "Driving the scalability of DNA-based information storage systems," *ACS Synth. Biol.*, vol. 8, no. 6, pp. 1241–1248, Jun. 2019.

[8] W. D. Chen, A. X. Kohll, B. H. Nguyen, J. Koch, R. Heckel, W. J. Stark, L. Ceze, K. Strauss, and R. N. Grass, "Combining data longevity with high storage capacity—Layer-by-layer DNA encapsulated in magnetic nanoparticles," *Adv. Funct. Mater.*, vol. 29, no. 28, Jul. 2019, Art. no. 1901672.

[9] C. N. Takahashi, B. H. Nguyen, K. Strauss, and L. Ceze, "Demonstration of end-to-end automation of DNA data storage," *Sci. Rep.*, vol. 9, Mar. 2019, Art. no. 4998.

[10] Y. Wang, M. Keith, A. Leyme, S. Bergelson, and M. Feschenko, "Monitoring long-term DNA storage via absolute copy number quantification by ddPCR," *Anal. Biochemistry*, vol. 583, Oct. 2019, Art. no. 113363.

[11] R. Heckel, G. Mikutis, and R. N. Grass, "A characterization of the DNA data storage channel," *Sci. Rep.*, vol. 9, Jul. 2019, Art. no. 9663.

[12] Y. Wang, M. Noor-A-Rahim, E. Gunawan, Y. L. Guan, and C. L. Poh, "Construction of bio-constrained code for DNA data storage," *IEEE Commun. Lett.*, vol. 23, no. 6, pp. 963–966, Jun. 2019, doi: 10.1109/lcomm.2019.2912572.

[13] D. Carmean, L. Ceze, G. Seelig, K. Stewart, K. Strauss, and M. Willsey, "DNA data storage and hybrid molecular–electronic computing," *Proc. IEEE*, vol. 107, no. 1, pp. 63–72, Jan. 2019, doi: 10.1109/jproc.2018.2875386.

[14] J. Davis, "Microvenus," *Art J.*, vol. 55, no. 1, pp. 70–74, 1996.

[15] F. Akram, I. U. Haq, H. Ali, and A. T. Laghari, "Trends to store digital data in DNA: An overview," *Mol. Biol. Rep.*, vol. 45, no. 5, pp. 1479–1490, Oct. 2018.

[16] S. Yazdi, R. Gabrys, and O. Milenkovic, "Portable and error-free DNA-based data storage," *Sci. Rep.*, vol. 7, Jul. 2017, Art. no. 5011.

[17] M. H. Garzon and R. J. Deaton, "Codeword design and information encoding in DNA ensembles," *Natural Comput.*, vol. 3, no. 3, pp. 253–292, Aug. 2004.

[18] M. G. Ross, C. Russ, M. Costello, A. Hollinger, N. J. Lennon, R. Hegarty, C. Nusbaum, and D. B. Jaffe, "Characterizing and measuring bias in sequence data," *Genome Biol.*, vol. 14, no. 5, p. R51, 2013.

[19] S. Yazdi, Y. B. Yuan, J. Ma, H. M. Zhao, and O. Milenkovic, "A rewritable, random-access DNA-based storage system," *Sci. Rep.*, vol. 5, Sep. 2015, Art. no. 14138.

[20] H. Hong, L. Wang, H. Ahmad, J. Li, Y. Yang, and C. Wu, "Construction of DNA codes by using algebraic number theory," *Finite Fields Appl.*, vol. 37, pp. 328–343, Jan. 2016.

[21] R. Gabrys, H. M. Kiah, and O. Milenkovic, "Asymmetric lee distance codes for DNA-based storage," *IEEE Trans. Inf. Theory*, vol. 63, no. 8, pp. 4982–4995, Aug. 2017, doi: 10.1109/tit.2017.2700847.

[22] W. Song, K. Cai, M. Zhang, and C. Yuen, "Codes with run-length and GC-content constraints for DNA-based data storage," *IEEE Commun. Lett.*, vol. 22, no. 10, pp. 2004–2007, Oct. 2018, doi: 10.1109/lcomm.2018.2866566.

[23] K. A. Schouhamer Immink and K. Cai, "Design of capacity-approaching constrained codes for DNA-based storage systems," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 224–227, Feb. 2018, doi: 10.1109/lcomm.2017.2775608.

[24] S. M. H. Tabatabaei Yazdi, H. M. Kiah, R. Gabrys, and O. Milenkovic, "Mutually uncorrelated primers for DNA-based data storage," *IEEE Trans. Inf. Theory*, vol. 64, no. 9, pp. 6283–6296, Sep. 2018, doi: 10.1109/tit.2018.2792488.

[25] D. C. Dai, D. Minic, and D. Stojkovic, "New wormhole solution in de Sitter space," *Phys. Rev. D, Part. Fields*, vol. 98, no. 12, p. 7, 2018.

[26] B. Wang, Q. Zhang, X. P. Wei, "Tabu variable neighborhood search for designing DNA barcodes," *IEEE Trans. Nanobiosci.*, vol. 19, no. 1, pp. 127–131, Sep. 2020.

[27] B. Wang, X. Zheng, S. Zhou, C. Zhou, X. Wei, Q. Zhang, and Z. Wei, "Constructing DNA barcode sets based on particle swarm optimization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 3, pp. 999–1002, May 2018, doi: 10.1109/tcbb.2017.2679004.

[28] D. Limbachiya, M. K. Gupta, and V. Aggarwal, "Family of constrained codes for archival DNA data storage," *IEEE Commun. Lett.*, vol. 22, no. 10, pp. 1972–1975, Oct. 2018, doi: 10.1109/lcomm.2018.2861867.

[29] J. Khoury, B. A. Ovrut, N. Seiberg, P. J. Steinhardt, and N. Turok, "From big crunch to big bang," 2002, *arXiv:hep-th/0108187*. [Online]. Available: https://arxiv.org/abs/hep-th/0108187

[30] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-verse optimizer: A nature-inspired algorithm for global optimization," *Neural Comput. Appl.*, vol. 27, no. 2, pp. 495–513, Feb. 2016.

[31] P. J. Steinhardt, "A cyclic model of the universe," *Science*, vol. 296, no. 5572, pp. 1436–1439, May 2002.

[32] S. R. Kane and D. M. Gelino, "The habitable zone and extreme planetary orbits," *Astrobiology*, vol. 12, no. 10, pp. 940–945, Oct. 2012.

[33] D. Steinley, "K-means clustering: A half-century synthesis," *Brit. J. Math. Stat. Psychol.*, vol. 59, no. 1, pp. 1–34, May 2006.

[34] K.-L. Chung and K.-S. Lin, "An efficient line symmetry-based K-means algorithm," *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 765–772, May 2006.

[35] Y. Terada, "Strong consistency of reduced *K*-means clustering," *Scand. J. Statist.*, vol. 41, no. 4, pp. 913–931, Dec. 2014.

[36] S. Shahrivari and S. Jalili, "Single-pass and linear-time k-means clustering based on MapReduce," *Inf. Syst.*, vol. 60, pp. 1–12, Aug. 2016.

[37] A. M. Bagirov, "Modified global-means algorithm for minimum sum-of-squares clustering problems," *Pattern Recognit.*, vol. 41, no. 10, pp. 3192–3199, Oct. 2008.

[38] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, Jun. 2010.

[39] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[40] J. Digalakis and K. Margaritis, "On benchmarking functions for genetic algorithms," *Int. J. Comput. Mathematics*, vol. 77, no. 4, pp. 481–506, Jan. 2001.

[41] X.-S. Yang, "Test problems in optimization," 2010, *arXiv:1008.0549*. [Online]. Available: http://arxiv.org/abs/1008.0549

[42] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[43] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, Jun. 2009.

[44] F. Vandenbergh and A. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Inf. Sci.*, vol. 176, no. 8, pp. 937–971, Apr. 2006.

[45] T. P. Hettmansperger, "The ranked-set sample sign test," *J. Nonparametric Statist.*, vol. 4, no. 3, pp. 263–270, Jan. 1995.

[46] D. H. Kim and Y. C. Kim, "Wilcoxon signed rank test using ranked-set sample," *Korean J. Com. Appl. Math.*, vol. 3, no. 2, pp. 235–243, Jun. 1996.

[47] D. Limbachiya, V. Dhameliya, M. Khakhar, and M. K. Gupta, "On optimal family of codes for archival DNA storage," *IWSDA, no.*, pp. 123–127,2015.

[48] B. Wang, Y. Xie, S. Zhou, X. Zheng, and C. Zhou, "Correcting errors in image encryption based on DNA coding," *Molecules*, vol. 23, no. 8, p. 1878, Jul. 2018.

[49] T. Song, A. Rodriguez-Paton, P. Zheng, and X. Zeng, "Spiking neural P systems with colored spikes," *IEEE Trans. Cogn. Dev. Syst.*, vol. 10, no. 4, pp. 1106–1115, Dec. 2018.

[50] C. Zhu, C. Zhou, and B. Wang, "Development of conceptual learning model based on various stability features," *IEEE Access*, vol. 7, pp. 37961–37969, 2019.

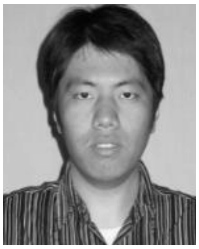
[51] T. Song, S. Pang, S. Hao, A. Rodríguez-Patón, and P. Zheng, "A parallel image skeletonizing method using spiking neural P systems with weights," *Neural Process. Lett.*, vol. 50, no. 2, pp. 1485–1502, Oct. 2019, doi: 10.1007/s11063-018-9947-9.

[52] T. Song, L. Pan, T. Wu, P. Zheng, M. L. D. Wong, and A. Rodriguez-Paton, "Spiking neural P systems with learning functions," *IEEE Trans. Nanobiosci.*, vol. 18, no. 2, pp. 176–190, Apr. 2019, doi: 10.1109/tnb.2019.2896981.

BEN CAO was born in Suzhou, Anhui, in 1997. He received the B.S. degree in computer science and technology from Huaibei Normal University, Anhui, in 2018. He is currently pursuing the master's degree in computer science and technology with Dalian University. His current research interests include intelligent algorithms, DNA storage, and DNA coding optimization.

SUE ZHAO received the B.S. degree in software engineering from Linyi University, Linyi, China, in 2017. She is currently pursuing the master's degree in computer technology with the Key Laboratory of Advanced Design and Intelligent Computing, Dalian University. Her current research interests include biological computing and intelligent computing.

XUE LI received the B.S. degree in engineering from Jining Medical College, in 2018. She is currently pursuing the master's degree in computer science and technology with the Key Laboratory of Advanced Design and Intelligent Computing, Dalian University. She is engaged in the direction of DNA coding optimization.



BIN WANG received the B.S. degree in computer science and technology from Dalian University, in June 2006, and the Ph.D. degree in mechanical design and theory from the Dalian University of Technology, in October 2013. He is currently an Associate Professor with Dalian University. His current research interests include intelligence computing, DNA sequence design, DNA cryptography, and biological networks. He has coauthored about 61 articles published.