IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 27, 2019, accepted June 13, 2019, date of publication June 24, 2019, date of current version July 22, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2924708

A BPSON Algorithm Applied to DNA Codes Design

KAIQIANG LIU[1], BIN WANG[1], HUI LV[1], XIAOPENG WEI[2], AND QIANG ZHANG[2]
[1]Key Laboratory of Advanced Design and Intelligence Computing, Ministry of Education, Dalian University, Dalian 116622, China
[2]School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

Corresponding author: Bin Wang (wangbinpaper@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61425002, Grant 61751203, Grant 61772100, Grant 61702070, Grant 61672121, Grant 61572093, and Grant 61802040, 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 in part by the High-level Talent Innovation Support Program of Dalian City under Grant 2017RQ060.

ABSTRACT DNA computing proposed by Adelman is new biotechnology which provides a new way to solve NP-hard problem. It has a promising future and successful results of various applications. DNA codes design is a significant step in DNA computing. Therefore, reliable DNA codes design not only can avoid non-specific hybridization between a code and its Watson–Crick complement but also can improve the efficiency of DNA computing. In this paper, a new algorithm is proposed to design reliable DNA codes. This algorithm combines the Bat algorithm and PSO algorithm. Fast nondominated sorting is used to assign a rank for codes. Thus, it is called BPSON for short. A bat algorithm is used to overcome PSO fall into the local optimal solution and enhance global search ability. In addition, for the purpose of verifying the effectiveness of our algorithm, the performance of BPSON is compared with the previous works. The experimental results show that codes obtained by our algorithm can avoid the appearance of secondary structure, which is beneficial to the specific hybridization among codes and has better thermodynamic properties. The results show that our algorithm can provide optimal codes for DNA computing.

INDEX TERMS BPSON algorithm, DNA computing, DNA codes design, fast nondominated sort.

I. INTRODUCTION

DNA computing is a parallel computing model based on DNA molecule. The core reaction of DNA computing is the hybridization reaction between DNA molecules. The accuracy of the reaction between molecules directly affects the results of DNA computing [1], [2]. DNA computing has the main advantages of high parallelism, massive information storage and low energy consumption [3]. Usually, DNA computing mainly includes three steps: firstly, the problem to be solved is mapped to a set of DNA molecules; second, carry out various biochemical reaction such as hybridization, connection, extension to generate possible solution space; finally, using PCR reaction to separate and read solution.

DNA codes design is a minimum optimization problem. The purpose of codes problem research in DNA computing is to obtain lower value of measure standards. The smaller value of measure standards, the better quality of the codes. Therefore, it is optimal codes. Optimal codes can reduce the probability of false hybridization as much as possible in the biochemical reaction process and improve the efficiency and reliability of DNA computing. DNA codes design is a vital step in DNA computing and it has a wide range of applications in bioinformatics research, such as DNA nanotechnology [4], DNA storage systems [5], DNA images encryption [6], DNA barcode [7]. Hybridization between a DNA code and its complement code is a significant step of DNA computing [8]. However, undesired hybridization often causes false computing. For this purpose, DNA codes need to be designed cautiously. In the next 10 years, although it is a promising computing model that can replace transistors and silicon-based computers, it has many technical shortcomings to overcome. In fact, when these technical problems are solved, it will become a significant area of computer engineering. Incorrect reactions are primarily false hybridizations. There are two types of false hybridization [9], [10]. One is the false positive, that is, a DNA molecule that is not fully complementary hybridizes under appropriate conditions

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

to form a double-stranded molecule; the other is the false negative, that is, a completely complementary DNA molecule does not hybridize during the course of the reaction for various reasons. To solve these shortcomings, many people devoted to improve the validity of DNA computing. An effective way is to design reliable DNA codes. The aim of design DNA codes is to reduce the false hybridization of DNA computing.

In order to achieve this aim, different algorithms are used to design DNA codes. Deep Learning [11] is very popular today, but it is used less in this area. The mainstream method of designing DNA codes is mainly intelligent computing. The simplest and most representative algorithm is exhaustive methods and random search algorithms [12], [13], but they are not very efficient because of using a lot of computer resources. Template mapping strategies [14], [15] were used to select dissimilar codes among numerous DNA codes. A directed graph is used by Feldkamp to design DNA codes [16]. In this method, graph nodes represent the basic chain, where each node has four chains, which can be shown as successors in a longer codes. Tanaka uses simulated annealing to generate DNA codes [17]. This method combined different constraints into a fitness function to find a reliable solution. Different from above algorithms, some evolutionary algorithms. [18] have been widely used recently. For example, genetic algorithms are often used to design DNA codes because of their simplicity [19], [20] and the fitness function takes Hamming distance into account. Zhang proposed an iterative genetic search to design codes in a DNA computing environment [21]. Arita combines similarity, H-measure, Hamming distance, and GC content as constraints of genetic algorithms [22]. Shin proposed a multi-objective evolutionary algorithm based on six constraints and improved the algorithm [23], [24]. The constraints are melting temperature and GC content, taking H-measure, continuity and Hairpin as the objective function. Other research work also considered different constraints, but they eventually turned the multi-objective problem into a single-objective problem by adding weights. Thus, Xu *et al.* [25] and Cui and Li [26] used a hybrid strategy to combine genetic algorithm and particle swarm optimization together as a solution to solve the multi-objective optimization problem of DNA codes design. Khalid *et al.* [27] uses PSO again to minimize the constraint H-measure. Kurniawan *et al.* [28], [29] used the ant colony optimization system to solve the single objective problem. On the other side, Shin *et al.* [24] used a multi-objective NSGA-II algorithm, they took several constraints into account and the results were fairly well, so our study used it as a comparison. Wang *et al.* [30] proposed INSGA-II to design DNA codes according to the standard NSGA-II. This novel algorithm introduces a constraint in the process of non-dominated sorting, compared with other algorithmic performances. This algorithm performs better. In any case, the above related work confirms the applicability of evolutionary methods.

In this paper, a hybrid bat algorithm based on fast non-dominated sorting was proposed to optimize DNA codes. The bat algorithm is a new bio-heuristic intelligent optimization algorithm proposed by Yang [31]. It simulates the echolocation behavior of bats in nature, constantly adjusting the search frequency during the search process, and speeding up the convergence speed of the algorithm. At the same time, the bat individual coordinates the exploration and mining of the algorithm by adjusting the variation of the acoustic pulse frequency and the impulse loudness, which enhances the search ability and robustness of the algorithm. Since its introduction, the bat algorithm has attracted the attention of many scholars. The standard bat algorithm has many advantages, such as simple and easy to implement model, potential parallelism and distributed, which can more flexibly balance the mining and exploration of the algorithm, and the convergence speed is fast. In our study, we use a hybrid strategy to mix the bat algorithm with the particle swarm optimization algorithm, and use the fast non-dominated sorting to calculate and rank the fitness of the codes. We make full use of the better global optimization ability of the bat algorithm to avoid the particle swarm optimization algorithm falling into the local optimal solution in the later stage, thus enhancing the diversity of the population and having strong global optimization ability. Our algorithm obtains a relatively stable melting temperature, and our algorithm obtains a small continuity, hairpin, similarity and free energy compared to the reference. In general, by comparison we can find that our proposed algorithm outperforms other algorithms.

The rest of the paper is structured as follows: In the second part, the constraints that DNA codes must be satisfied and the multi-objective formula are described. The third part describes the related heuristic optimization algorithm. The fourth part is the analysis of the results. The final part is the summary of this paper.

II. DNA CODES DESIGN

DNA codes design usually takes many constraints into account. So, it is a multi-objective optimization problem. The traditional way to deal with multi-objective problems is to assign different constraints to certain weights and then add them to a single objective problem. But this method is too subjective in the selection of weights. Therefore, this paper adopts the method of fast non-dominated sorting of classical algorithms for dealing with multi-objective problems. Generally speaking, the more constraints are considered, the more reliable the codes is obtained, but the inclusion of a large number of constraints is computationally impractical and some constraints overlap each other. Therefore, in this paper, we have selected six biochemical standards as constraints. They are continuity, hairpin structure, H-measure, similarity, melting temperature, GC content. Low Similarity and H-measure can avoid non-specific hybridization, low continuity and hairpin structure can avoid undesired secondary structure, fixed GC content and melting temperature can maintain stable thermodynamic properties [24].

A. CONTINUITY

Continuity represents the same bases continuous appear in an oligonucleotide. This constraint usually given a threshold, if the number of continuous same bases bigger than given threshold, we take it for violating continuity constraints [32]. For instance, assume that 4 is the threshold, a code CCATTCGCTCCCCC, the third part with underline contains 5 bases C, violates the continuity threshold. The mathematical formula is:

$$f_{continuity}(\sum) = \sum_{i=1}^{n} Continuity(\sum_i) \tag{1}$$

$$\text{Continuity (x)} = \sum_{i=1}^{l-t+1} \sum_{a\in\{\text{A,G,C,T}\}} T(c_a\,(x,i)\,,t)^2 \tag{2}$$

where $\sum_i$ denotes a DNA code from DNA codes set $\sum$ and n is the number of codes in the DNA set $\sum$. Letter t is the continuity threshold, l is the total number of bases in x. T(C_a (x, i) , t) is a threshold function which returns $C_a(x, i)$, if $C_a(x, i)$ >t and 0 otherwise. If $\exists\varepsilon$, s. t. $x_i \neq a, x_{i+j} = a(1 \leq j \leq \varepsilon, x_{i+e+1} \neq a)$, c_a (x, i) = ε, Cotherwise is 0.

B. HAIRPIN

In an oligonucleotide DNA molecule self-hybridization formed Hairpin structure. It mainly studies the probability that a DNA code can produce a secondary structure. We assumed that the hairpin has a loop and a stem, a loop contains at least R_{min} bases and a stem contains P_{min} bases pairs. The formula (3) and (4) consider that in a code x the hairpin stem having the r base loop and the p base pair is formed at the position i. The mathematical expression [32] is:

$$f_{Hairpin}(\sum) = \sum_{i=1}^{n} Hairpin(\sum_i) \tag{3}$$

$$\text{Hairpin (x)} = \sum_{p=P_{min}}^{l-R_{min}/2} \sum_{r=R_{min}}^{l-2p} \Sigma_{i=1}^{l-2p-r} \text{T}(\sum_{j=1}^{pinlen(p,r,i)} bp\left(x_{p+i+j}, x_{p+i+j+r}\right), \frac{pinlen(p,r,i)}{2}) \tag{4}$$

where n is the number of codes in the DNA set $\sum$, pinlen (p, r, i) = min(p + i,l − r − i − p) indicates the number of possible bases required to form a hairpin at centre p + i + r/2. bp $\left(x_{p+i+j}, x_{p+i+j+r}\right)$ is equal to 1 if x_{p+i+j} and $x_{p+i+j+r}$ are complementary bases and 0 otherwise.

C. SIMILARITY

This constraint mainly refers to the similarity degree of two codes and the similarity degree of shift position of two given codes in the same direction. It primarily is aimed to make each code not similar with other code. It generally is divided into two aspects: Continuous and discontinuous similarities. For example, AGTGTAATGGGG andACGCCTATGGGA have the same continuity 5 from the 7th base to the 11th base is subcode ATGGG. The discontinuous similarity is 6. The mathematical definition of this measurement [33] is:

$$f_{sim}\,(\Sigma) = \sum_{i=1}^{n} \sum_{j=1}^{n} Sim(\Sigma_i\Sigma_j) \tag{5}$$

where Σ_i and Σ_j are parallel DNA codes in Σ, n is the number of codes in the DNA set Σ. In addition , $Sim(\Sigma_i\Sigma_j)$ is divided into two terms. One is for the overall discrete discontinuous similarity, and the other is the penalty for the continuous common subcode. Formula as following:

$$\text{Sim (x, y)} = Max_{g,i}(s_{disc}\left(x, shift\left(y\,(-)^g\,y, i\right)\right) + s_{cont}(x, shift(y\,(-)^g\,y, i))) \tag{6}$$

where shift indicates that the code y is offset by bases of i, and (-) represents a gap , 0 ≤ g ≤ −3, the code length represented by letter l and $|i| \leq l$.

$$s_{disc}\,(x, y) = T(\sum_{i=1}^{l} eq\,(x_i, y_i)\,, s_{disc} \times length_{nb}(y)) \tag{7}$$

$$s_{cont}\,(x, y) = \sum_{i=1}^{l} T(ceq\,(x, y, i)\,, s_{cont}) \tag{8}$$

where the value of s_{disc} is between 0 and 1; s_{cont} is an integer between 1 and l; ceq(x, y, i) measures the length of a continuous common subcode starting from the ith base of code x. If $x_i = y_i$, eq(x_i, y_i) is equal to 1, otherwise equal to 0. Finally, if i > j, T(i, j) is equal to i, otherwise is 0.

D. H-MEASURE

H-measure is used to avoid cross-hybridization between codes. It is a very important constraint. H-measure is used to calculate how many nucleotides are complementary between the codes in a set. The mathematical definition is described in Eq. (9).

$$f_{H-measure}(\Sigma) = \sum_{i=1}^{n} \sum_{j=1}^{n} H - measure(\Sigma_i\Sigma_j) \tag{9}$$

where Σ_i and Σ_j are anti-parallel codes. We use letter n to indicate the number of codes which are in the set Σ. Moreover, there are two parts about H-measure. One is the penalty for the continuous complementary region and the other for the overall complementarily. The measure is:

$$H_{measure(x,y)} = Max_{g,i}(h_{disc}\left(x, shift\left(y\,(-)^g\,y, i\right)\right) + h_{cont}(x, shift\left(y\,(-)^g\,y, i\right))) \tag{10}$$

where shift denotes a shift of i bases in a code y. Symbol (-) denotes a gap, $|i| \leq l$, and 0 ≤ g ≤ 1 − 3.l denotes the length of codes. $y(-)^g y$ denotes g gaps between code y and its complement code.

$$h_{disc}(\text{x, y}) = T(\sum_{i=1}^{l} bp\,(x_i, y_i)\,, H_{disc} \times length_{nb}\,(y)) \tag{11}$$

$$h_{cont}\,(x, y) = \sum_{i=1}^{l} T(cbp\,(x, y, i)\,, H_{cont}) \tag{12}$$

In the above formula, H_{disc} is a real value between 0 and 1; H_{cont} is an integer between 1 and l; cbp(x, y, i) denotes the length of continuous base from the position of i of code x; bp(x_i, y_i) is equal to 1 if $x_i = \bar{y_i}$ and equal to 0 otherwise. Finally, if i>j T(i,j) = i and 0 otherwise.

E. GC CONTENT

This constraint study the percentage of bases C and G in an oligonucleotide. It is one of the thermodynamic properties. For example, code CC TTAGAACGTCTCGAGCTT the GC% is 50%. Since GC contains three hydrogen bonds and AT contains two hydrogen bonds, the base pair releases more energy when the GC is destroyed. Therefore, the level of GC indirectly affects the melting temperature.

F. MELTING TEMPERATURE

Melting temperature of a DNA is a significant parameter of Primer. During the denaturation of DNA molecules, the double-stranded DNA molecules undergo physical changes. The process of double-stranded into a single-strand will release the temperature, and the temperature released by half of the molecules will become the melting temperature. There are many ways to calculate the melting temperature, and GC% technology [34] and the nearest neighbor model [35] are widely used. In our research work, we use the nearest neighbor model to calculate the melting temperature.

G. FREE ENERGY

It is used to predict thermodynamic properties of DNA precisely for molecular biology experiments. Free energy constraint is mainly used to check stability of hybridization between two codes. In this paper, we used nearest neighbor model [35] to calculate Free energy for DNA codes. The formula of Free energy as following:

$$\Delta \mathrm{G}^{\circ}(DNA) = \sum\nolimits_i n_i \Delta G^{\circ}(i) + \Delta G^{\circ}(Init.W\mathrm{lterm}.G.C) + \Delta G^{\circ}(Init.W\mathrm{lterm}.A.T) + \Delta G^{\circ}(sym) \quad (13)$$

Assumption that code is self-complementing, $\Delta G^{\circ}(sym) = 0.43$, otherwise, $\Delta G^{\circ}(sym) = 0$.

H. OTHER RECOMMENDATIONS

DNA codes design problem is a multiobjective optimization problem using our proposed BPSON Algorithm. It needs to consider four objects are similarities, H-measure, continuity and hairpin structure and two constraints (melting temperature and GC content).The formula is expressed as follows:

$$\begin{aligned} &\text{Minimize } f_i(x), \\ &\text{where i} \in \{\text{continuity, hairpin, H} - \text{measure, similarity}\} \\ &\text{subject to the constraints : } T_m \text{ and } GC \text{ content} \end{aligned} \quad (14)$$

III. DESCRIPTION OF THE ALGORITHMS

A. BAT ALGORITHM

The Bat Algorithm is a new intelligent optimization algorithm founded by Xin-she Yang in 2010. [31]. It simulates the echolocation behavior of bats in nature. They use echolocation methods to capture prey or avoid obstacles. Bats produce a very loud sound wave when flying, and constantly receive sound waves reflected from around the object. The pulse from the bat is related to the hunting strategy, which also depends on the species. Most bats emit a different frequency of sound waves while scanning, while others use only constant frequencies, and their sound frequency depends on the type of bat. The behavior of bats that emit sonic pulses can be defined as an optimized form of the objective function, inspired by this principle to produce the bat algorithm. It uses the ultrasonic features of the miniature bat to develop a new heuristic algorithm, the bat algorithm, which is based on the following ideal rules:

(1) All bats judge the difference between food and obstacles using the method of echolocation;

(2) Each bat at the position x_i performs a random flight at the speed v_i and takes a different wavelength (or frequency f_i) and a sound intensity A_i to search for the prey. They automatically adjust the wavelength (or frequency) of the emitted pulse according to the degree of proximity to the target;

(3) Although the pulse loudness varies differently in different forms, it is assumed here that the loudness decreases as the algebra increases, from the maximum value A_{max} to the minimum value A_{min}.

In addition to the above guidelines, for the sake of simplicity, some approximations are used in the bat algorithm, such as the range of the frequency f is $[f_{min}, f_{max}]$, and the range of the corresponding wavelength is $[\lambda_{min}, \lambda_{max}]$. This adjusts the range of the search by adjusting the frequency to search for the area of interest. Let the definition domain be the D-dimensional search space. At t + 1, the speed and position update method of bat i is:

$$v_i^{t+1} = v_i^t + \left(x_i^t - x^*\right) \cdot f_i \quad (15)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (16)$$

where v_i^t, v_i^{t+1} respectively represent the flight speeds of bat individual i at t and t + 1; x_i^t, x_i^{t+1} respectively represent the location bat ith individual at t and t + 1; x^* represents the global optimal position. The frequency f_i is the pulse frequency of the bat individual i during the search, which is defined as follows:

$$f_i = f_{min} + (f_{max} - f_{min}) \cdot \beta \quad (17)$$

where β is a number between [0, 1], f_{min} is the lower and f_{max} is the upper limits of the pulse frequency. In order to make the algorithm have better search performance, the bat algorithm designed the following local search methods:

$$x_i^t = x^* + \varepsilon A^t \quad (18)$$

where ε is a random number between $[-1, 1]$ and A^t is the average loudness of the bat at time t.

It is known from the biological knowledge that the vocal pulse emitted by the bat in the early stage of searching for prey has a strong sound intensity and a high frequency, which can be searched in a large range. When the prey is found, the pulse intensity is reduced. Increase the pulse frequency to more accurately locate the prey position. (19) and (20) are

used to simulate this search process.

$$r_i^{t+1} = r_i^0[1 - \exp(-)\gamma t] \tag{19}$$

$$A_i^{t+1} = \alpha A_i^t \tag{20}$$

where r_i^0 is the maximum pulse frequency, r_i^{t+1} is the pulse frequency of the bat at time t + 1; A_i^t, A_i^{t+1} are the pulse sound strengths emitted at times t and t + 1, respectively; γ and α are the pulse frequency increase coefficient and the pulse loudness attenuation coefficient, respectively, which are constants greater than zero.

B. PARTICLE SWARM OPTIMIZATION ALGORITHM

Kennedy and Eberhart proposed PSO algorithm in 1995 [36], [37]. The idea of this algorithm is inspired by the behavior of birds. By studying the collective behavior of birds, researchers find that there is an information sharing mechanism among bird populations, which is the basis for the formation of PSO algorithm. A particle is a solution to a solution space. The best position for each particle is the best solution for the particle. The best position of the entire population is the global optimal solution of the algorithm. We use pBest to represent the best solution for a particle and gBest to represent global optimal solution. Fitness function can be used to measure the quality of particles. Each particle generates a new generation of population by constantly updating pBeat and gBest. N is the population size, X_i represents the position of ith (i = 1, 2, …, N) particle, and pBest[i] represents the best optimal solution of individual i. V_i is the velocity. The index of the best particles in the group is represented by the symbol g, according to formula. [38] position and velocity is updated:

$$V_i = w \cdot V_i + c1 \cdot rand() \cdot (pBest\,[i] - X_i) + c2 \cdot Rand \cdot (pBest\,[g] - \mathrm{X_i}) \tag{21}$$

$$X_i = X_i + V_i \tag{22}$$

where the learning factors are c1 and c2; two random numbers respectively are represented by rand and Rand ; inertia weight is represented by the letter w. The formula consists of three parts. The first part is the current state of the particle, mainly previous speed. It can balance search ability between local and global search; We usually call the second part the cognitive part, it can avoid local optimal and enhance global optimization ability; The social part is the last part about information sharing. The three parts work together to achieve the best position.

C. FAST NON-DOMINATED SORT

It is currently considered to be the most efficient to deal with multi-objective problems is NSGA-II with elite strategy [39]. The core of the algorithm is the fast non-dominated sorting strategy. The algorithm reduces the problem to a fitness function by non-dominated classification.

The basic idea of the algorithm is to randomly initialize a population with a size of N. After a series of genetic operations, the next generation of subpopulations is obtained, and the subpopulations are merged with the parent population, and the combined individuals are used as the second generation population. The population is once again subjected to fast non-dominated sorting, and the same sorted individuals in the fast non-dominated sorting are ranked by crowding degree. The next generation of individual is determined according to the sort order and the degree of crowding of the population.

The process of fast non-dominated sorting is elaborating as following: Each individual p needs to be defined with two attributes: the number of n_p that dominates the individual p in the population P_t and the set S_p consisting of all individuals p dominated. Where p is a random integer that does not exceed the size of the population. First, initialize $n_p = 0$, $S_p = \emptyset$. Next, traverse all individuals in the population and calculate n_p and S_p for each individual. The specific calculation method is as follows: suppose p, q are any two individuals in the population. If p dominates q, then $S_P = S_p \cup \{q\}$, $n_q = n_q + 1$; if q dominates p, then $S_q = S_q \cup \{p\}$, $n_p = n_p + 1$. Then, the n_p and S_p results obtained by the calculation are layered, the non-dominated layer number is initialized to 1. First, find all individuals with $n_p = 0$ in the population, remove them from P_t and divide them into the set of current non-dominated hierarchical on the non-dominated sorting hierarchy; then the n_p value of the corresponding individual S_p in these individuals is subtracted by 1 and the hierarchical ordinal number is recursively increased by 1. This is repeated until all individuals are divided into non-dominant sets.

This paper determines the pros and cons of individuals based on the division of non-dominated ranks, and assigns ranks to non-dominated solutions. Individuals with higher ranks are superior to individuals with lower ranks. In this paper, the Sigmoid activation function of artificial neural network [40] is used to assign fitness. The individual fitness formula is as follows:

$$\mathrm{f} = \frac{1}{1 + e^{\frac{r(i)-1}{R}}} \tag{23}$$

where r(i) is the number of ranks the individual has assigned in the population and R is the maximum number of ranks of the current population.

D. BPSON ALGORITHM

This paper proposes a BPSON algorithm combined Bat algorithm and particle swarm optimization algorithm together, using fast non-dominated sorting to optimize DNA codes. The bat algorithm is more suitable for dealing with low-dimensional problems [41]. In order to make up for the shortcomings of this algorithm, this paper will combine the bat algorithm and the particle swarm algorithm together to make the bat algorithm able to deal with higher dimensional problems. However, parameters of a standard PSO decide its performance and it's easy to get caught up in local optimization [42]. The resulting solution also has some random features. Our proposed algorithm can avoid these problems. In the early, PSO was used to optimize the DNA codes obtained from the initial population, and then the

bat algorithm was used to further optimize or abandon the codes that not satisfied continuity, hairpin, H-measure and similarity. The frequency adjustment mechanism of the bat algorithm can increase population diversity and have better global optimization ability. This algorithm firstly uses the particle swarm optimization algorithm to search the initial population to obtain the optimal codes set; Secondly, the fitness calculation and ordering of the optimal codes obtained by the particle swarm are performed by fast non-dominated sorting, it can assign rank for codes, and the lower-ranking codes are less likely to be excluded from the next generation, higher ranked codes will be saved and will continue to operate in the next generation; Then, the codes obtained after sorting is further optimized and adjusted by the bat algorithm. The bat algorithm adjusts the codes that is not satisfied the constraint obtained by the particle swarm optimization algorithm through the update of velocity and position, so that it evolves to the direction with the smallest fitness value, the bat algorithm will abandon some unqualified boundary values from the particle swarm algorithm and use the preservation strategy to retain the optimal value; Finally, the fitness of the codes are calculated by fast nondominated sort again and the codes with better quality are selected according to the fitness order.

In this study, the mapping between numbers and bases is as follows: F0-C, 1-T, 2-A, 3-G.

The detailed steps are as follows:

Step1: Initialize the population and the parameters required by the algorithm;

Step2: Search the initial population with the particle swarm algorithm, the velocity and position of the particles are constantly updated through iteration, record the global optimal value and the individual extremum each time. By tracking these two extreme values, the particle swarm algorithm can search the Optimal solution;

Step3: The optimal codes set searched by the PSO is used as the input of the bat algorithm, and the bat algorithm is used for further optimization;

Step4: For each bat individual, generate a random number Rand1, determine the size of Rand1 and the pulse frequency r, if Rand1 is greater than the pulse frequency r, then perform step 5, otherwise perform step 6;

Step5: Re-disturbing the generation near the current optimal individual;

Step6: Calculate the fitness of bat individual at new position;

Step7: Determine the size of the random number Rand2 and the intensity A generated by each bat individual and the size of each individual fitness value and the global optimal value. If Rand2$<$A and the individual fitness value is less than the global optimal value, execute Step 8, otherwise perform step 9;

Step8: Take the current individual as the optimal solution, reduce the sound intensity, and increase the pulse transmission frequency;

Step9: Sort the fitness values to update the current optimal solution;

Step10: Determine whether the maximum number of iterations is reached, if step 11, otherwise, return to step 2;

Step11: Sort the results of the results, and output the sorted results. Fig.1. is the flow chart of this algorithm.

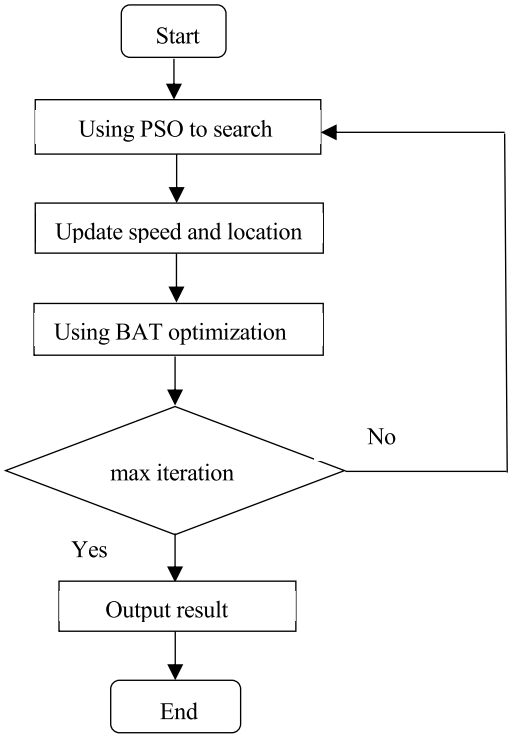

FIGURE 1. Flow chart of the algorithm.

IV. RESULTS AND ANALYSIS

A. ALGORITHM PARAMETER

In this paper, the algorithm is simulated by MATLAB R2017a under Win7 Intel(R) CPU3.6GHz, 4.0GB RAM. All parameters in TABLE 1 affect the experimental result. We have performed 16 of experiments using 0.1 $\sim$ 0.9 as the value for last 4 parameters in TABLE 1. By comparing the results of each experiment, we only show the parameters setting that gives the optimal result of our proposed algorithm in TABLE 1.

TABLE 1. Parameters used in our algorithm.

Symbol	Meaning	Value
PSO_ini	Population size	20
iter_max	Number of iterations	200
r	Bat algorithm transmission frequency	0.5
A	Bat algorithm pulse loudness	0.5
gamma	Pulse frequency increase factor	0.5
alpha	Impulse attenuation coefficient	0.25

B. RESULTS AND ANALYSIS

This research problem can be seen as a minimum optimization problem. The lower the continuity, the hairpin structure,

TABLE 2. Comparison of the codes provided by INSGA-II and codes obtained by our algorithm for the data set with 10 codes and 20 nucleotides per code.

DNA codes (5'-3')	Continuity	Hairpin	H-measure	Similarity	Tm	GC%
Our codes						
CCTTAGAACGTCTCGAGCTT	0	0	88	78	63.81	50
CCGTATACATGCTCGGTCTT	0	0	86	81	63.97	50
ACTCGCGACTACCAACATCT	0	0	98	75	65.67	50
CGTATCTGTGCTTCTCGTCA	0	0	92	84	64.13	50
ATTGCTGGATGAGGTGCCTA	0	0	94	80	65.84	50
TTCACTACTGAGGATCCGCA	0	0	98	76	65.09	50
CCTTAGAACGTCTCGAGCTT	0	0	88	78	63.81	50
CCGTATACATGCTCGGTCTT	0	0	86	81	63.97	50
ACTCGCGACTACCAACATCT	0	0	98	75	65.67	50
TTCACTACTGAGGATCCGCA	0	0	98	76	65.09	50
INSGA-II[30]						
CGTCTAGGCCGGATCAATAT	0	3	91	81	63.50	50
GGTTGTCCTGAGTGTTGTGT	0	0	84	80	64.81	50
AGAGTCAGCAGCGTAGAGAT	0	0	93	82	64.84	50
TACAGTCGGTTCGGTTATGG	0	0	90	73	63.84	50
GCGGAAGTAATCGGAAGTGA	0	0	88	81	64.46	50
ACGCCACAGTATATCATCGC	0	3	82	78	64.65	50
AGTCATTCTCCTGGCATTGC	0	6	90	76	65.12	50
GCACTGGTCAAGACATGGAT	0	3	94	81	64.76	50
TACACGTAGCCTGAACCATC	0	0	78	79	63.85	50
GCATCCTGTACTAACGTGGT	0	6	91	83	64.13	50

the similarity and the H-measure, the better [24]. The more stable the GC content and the melting temperature, the better. The smaller the free energy of the DNA codes, the better the hybridization reaction. Low continuity and hairpin structure can effectively avoid the generation of secondary structures that hinder DNA computing. Low similarity and H-measure can avoid non-specific hybridization between codes that control mismatch. For the thermodynamic properties of GC content and melting temperature, controlling it can make the experimental reaction more precise. Smaller free energy is fit to the occurrence of hybridization reactions.

In this part, we performed more than 50 simulations via development environment MATLAB. We compared our codes with the codes obtained by chaos particle swarm optimization algorithm [43], the codes from shin used NSGA-II [24] which is classical literature in the field of codes optimization, the codes obtained by Xiao's proposed quantum chaotic particle swarm evolution algorithm [44], the codes obtained by Invasive weed optimization algorithm [45] and the codes obtained by Dynamic membrane evolution algorithm [46]. All of the above comparisons use seven codes, 20 bases per code. In addition to, we use ten codes, 20 bases per code compared with the latest research results from Wang proposed improvement nondominated sort genetic algorithm-II [30]. We select seven or ten codes from our result through writing a MATLAB function. The codes for comparison is taken from the experimental results section of comparative literature. The range of Continuity and hairpin are from 0 to 9, most of the time it is 0. Occasionally, one or two other values appear. However, the range of H-measure and similarity are from 37 to 98. If we put continuity and hairpin as the priority, it is easy to obtain low values for four measure standards. If we put H-measure and similarity as the priority, it is not easy to ensure the H-measure and similarity are low meanwhile the continuity and hairpin are all 0 value. Therefore, ordering the Continuity/hairpin as the priority is helpful to make the whole four measure standards obtain low values.

Table 2 is a comparison between the results obtained by our improved algorithm and those obtained by INSGA-II algorithm from six aspects: Continuity, Hairpin structure, H-measure, similarity, melting temperature and GC content. Both we and the comparison algorithm selected 10 codes, each code containing 20 bases. To ensure fairness in the experiment, biochemical restrictions need to be adjusted to make the parameters the same. Therefore, at least six base stems and six base loops are required to form a hairpin structure. The lower limit of H-measurement and similarity is 6 bases in the case of continuous, and 0.17 in the case of discontinuity. The threshold for continuity is 2. In addition, we recalculated the melting temperature and free energy using the nearest neighbor model with a DNA concentration of 10nM and a salt concentration of 1M. H-measure and similarity are measures calculated between two codes. To ensure that our results are consistent with our definitions in Section II. We write a MATLAB program according to definitions in Section II. So, in the result, each code has its own similarity and H-measure.

Fig. 2 is the result of comparison between the codes obtained by our algorithm and that obtained by INSGA-II in terms of Continuity and Hairpin. The y-axis is the average fitness value.

Fig.3 is the result of comparison between the codes obtained by our algorithm and that obtained by INSGA-II in terms of H-measure and Similarity. The y-axis is the average fitness value.

TABLE 3. Comparison of the codes provided by CPSO, NACST/Seq, QCSEA, IWO, DMEA and codes obtained by our algorithm for the data set with 7 codes and 20 nucleotides per code.

DNA codes (5'-3')	Continuity	Hairpin	H-measure	Similarity	Tm	GC%
Our codes						
CCTCCTCCGCTCAATTGTAA	0	0	56	47	64.17	50
ACTCGATGACCTAGAGCCTA	0	0	66	56	63.64	50
CATACCTTATATCTCGGCCG	0	0	66	45	61.73	50
GGACTCTTGTTCGTTGCTGT	0	0	61	44	65.17	50
GGACTCTTGTTCGTTGCTGT	0	0	61	44	65.17	50
CATACCTTATATCTCGGCCG	0	0	66	45	61.73	50
CCTCCTCCGCTCAATTGTAA	0	0	56	47	64.17	50
CPSO[43]						
GACCGGTAAGATGAAGAGGA	0	0	60	50	62.94	50
CTATGCTTCTATCGCCTTCC	0	0	61	51	62.23	50
TAGTTGCACGAGAGAAGCAG	0	0	60	51	64.38	50
CGTGTACGAGCCTAATAAGG	0	0	64	54	62.14	50
CTTTGTCCATTGCACATCCG	9	0	61	53	64.42	50
TCCTATCCGAGATGATCCGT	0	3	63	55	64.08	50
TTCAACTTACGCTGTACGGC	0	6	63	54	62.25	50
NACST/Seq[24]						
CTCTTCATCCACCTCTTCTC	0	0	43	58	61.38	50
CTCTCATCTCTCCGTTCTTC	0	0	37	58	61.44	50
TATCCTGTGGTGTCCTTCCT	0	0	45	57	64.46	50
ATTCTGTTCCGTTGCGTGTC	0	0	52	56	65.83	50
TCTCTTACGTTGGTTGGCTG	0	0	51	53	64.63	50
GTATTCCAAGCGTCCGTGTT	0	0	55	49	65.30	50
AAACCTCCACCAACACACCA	9	0	55	43	66.71	50
QCSEA[44]						
CCATCTGCTTCACCGATTTA	9	3	65	51	62.33	45
AGTGCAGTACCGAGAATATT	0	0	67	51	60.73	40
ATTGAGCGCCCGGACTTCTC	9	0	64	56	69.43	60
GATTGCGAGAAGGTGTGGAT	0	0	58	55	64.79	50
GGGTGTAGAGTAGTCTCAGA	9	0	63	58	61.62	50
CGTGTTCCTATTCCTTGTCC	0	0	57	54	62.60	50
TAGTCTCTAACTCGGTTGTC	0	0	62	55	60.36	45
IWO[48]						
GATGGATTTACCTTGCACCT	9	4	60	54	62.29	45
CCTTCTCTCGTCTTCATACA	0	0	61	53	60.72	45
ACGATCGATTAATGGGAGTC	9	3	66	50	61.52	45
ATAAGTAGGGACTGCTCTAC	9	0	68	52	59.84	45
CCTAAGAACACAGGGCATAG	9	4	65	53	62.04	50
CTGGAAGCGTTTGCTAACTT	9	6	66	52	63.38	45
GCAGATTCCCGGATACTCAG	9	7	66	56	64.34	55
DMEA[49]						
TGAGTTGGAACTTGGCGGAA	0	0	70	52	66.76	50
CAGCATGTTAGCCAGTACGA	0	0	60	55	64.65	50
TTGAGTCCGCGTGGTTGGTC	0	0	63	53	69.79	60
AATTGACACTCTGATTCCGC	0	0	73	58	62.89	45
CATACATTGCATCAACGGCG	0	0	67	53	64.84	50
ATACACGCACCTAGCCACAC	0	0	59	50	66.93	55
GTTCCACAACAGGTCTAATG	0	3	61	53	60.65	45

FIGURE 2. Average fitness of continuity and hairpin for BPSON and INSGA-II.

It can be clearly seen from the bar charts of Fig.2 and Fig.3 that our algorithm achieves the lowest value 0 in terms of continuity and hairpin structure. In terms of continuity, the value both our algorithm and INSGA-II are 0. The value of the Hairpin structure obtained by our algorithm is far less than that obtained by INSGA-II algorithm. By comparison, we can conclude that our codes can prevent the generation of secondary structure. Meanwhile, codes obtained by our algorithm has a smaller Similarity value 78.4 compared to INSAGA-II. In general, our algorithm can avoid the occurrence of non-specific hybridization.

As Shown in Table 3, the codes obtained by our algorithm is compared with the codes obtained by other intelligence optimization algorithm. In table 3 we took seven codes, each code contains 20 bases. It is found that the proposed algorithm achieves better results than CPSO, QCSEA, NACST/Seq, IWO and DMEA in both continuity and hairpin structure. This shows that our codes can effectively avoid secondary

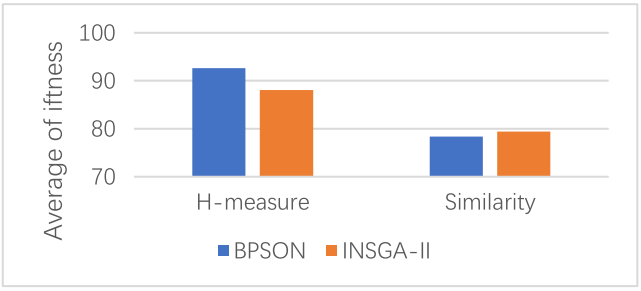

FIGURE 3. Average fitness of H-measure and Similarity for BPSON and INSGA-II.

structure generation. In terms of similarity, our results are far superior to other algorithms, indicating that our algorithm can effectively control non-specific hybridization, but our algorithm performs the same as CPSO on H-measure, better than QCSEA, IWO, DMEA and slightly higher than NACST/Seq. Overall, non-specific hybridization between codes can be effectively avoided. The GC content was always maintained at 50%, indicating that the codes generated by the algorithm has more stable thermodynamic properties.

TABLE 4. Comparison results of the melting temperature and free energy with 7 codes and 20 nucleotides per code.

	BPSON	CPSO[43]	NACST/Seq[24]	QCSEA[44]	IWO[45]	DMEA[47]
Var	2.09	1.44	4.33	9.85	2.33	8.83
ΔG°_{37}	-32.31	-32.39	-32.36	-31.83	-31.13	-32.91

TABLE 5. Comparison results of the melting temperature and free energy with 10 codes and 20 nucleotides per code.

	BPSON	INSGA-II[30]
Var	0.72	0.29
ΔG°_{37}	-32.83	-32.34

Table 4 and 5 are comparison results of the melting temperature and free energy that required by biochemical reaction to proceed. In Table.4 and Table.5, "Var" indicates the variance of melting temperature, "ΔG°_{37}" indicates the average free energy. In generally, biochemical reaction needs a relatively stable temperature and smaller free energy. The variance is usually used to measure the stability of a property, so we calculated the variance of melting temperature. As shown in Table 4 and 5, the variance of the codes obtained by the algorithm we proposed is smaller than all algorithms apart from CPSO and INSGA-II. It indicates that the melting temperature of the codes obtained by our algorithm is relatively stable, which is conducive to the accurate reaction of biochemical reaction.

In Table 4 and Table 5, ΔG°_{37} is the average free energy of codes in each generated code set. The free energy of each code is calculated by the code and its reverse complementary pair.

The value is a negative number, and a larger negative number indicates a smaller free energy. By comparison, it is found that the free energy of the codes obtained by our algorithm is better than INSGA-II and IWO, but a little higher than other algorithms. Free energy is calculated between two DNA codes for indicating the potential of their hybridization, the lower, the more possible. The minimum free energy (MFE) is widely used term calculated by one DNA code with its complementary pair. In Table 4 and Table 5, the lower value shows that the codes included in the set are easier to hybridize with their reverse complementary. Whether melting temperature, GC content or free energy, the codes obtained by our algorithm exhibits very good thermodynamic properties.

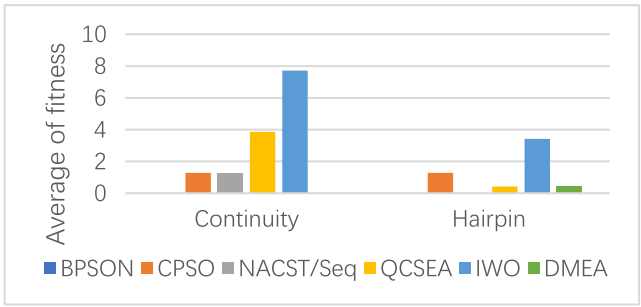

FIGURE 4. Average fitness of continuity and hairpin for BPSON and other algorithms.

Fig. 4 shows the average continuity and hairpin structure of seven strands containing 20 nucleotide strands obtained by our algorithm and other algorithms. The bar chart clearly shows that the codes obtained by our algorithm is superior to other algorithms in terms of continuity and hairpin structure, and the minimum value is obtained. Therefore, our algorithm can avoid the generation of secondary structure.

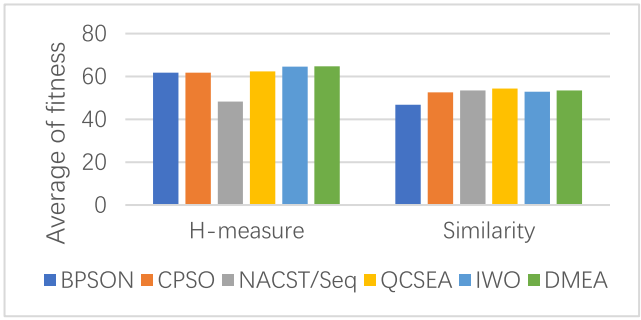

FIGURE 5. Average fitness of H-measure and Similarity for BPSON and other algorithms.

Fig. 5 show the average fitness values of our algorithm and other algorithms for H-measure and similarity. The bar chart clearly shows that the codes obtained by our proposed algorithm have the lowest similarity 46.9. In addition, our algorithm and CPSO algorithm have the same performance on H-measure 61.7, which is better than the performance of other algorithm, but slightly higher than

NASCT/Seq 48.28. Lower H-measure and similarity have bigger probability of hybridization between codes and its Watson-Crick complementary codes, thereby avoiding the occurrence of non-specific hybridization. Generally speaking, codes obtained by our algorithm are fit to specific hybridization.

V. CONCLUSION

In this paper, we proposed a BPSON algorithm based on non-dominated sorting to design reliable DNA codes that can be used for molecular computing. The proposed algorithm combines the particle swarm optimization algorithm with the bat algorithm to make full use of the fast convergence and global search ability of the bat algorithm to avoid the late arrival of the particle swarm optimization algorithm into the local optimal solution. Thereby improving the global search ability of the algorithm and ensuring the diversity of the population. We use the classical algorithm NSGA-II dealing with multi-objective problems to perform fast non-dominated sorting of codes, which helps us find the optimal encoding. We compare the codes obtained with other research work to find the smallest similarity, continuity and hairpin structure. The melting temperature of our codes are also relatively stable, and the GC content is strictly controlled at 50% to obtain a stable DNA codes. In the future our research work will focus on reducing the value of H-measure, making our codes hybridization reactions more precise and avoiding undesired non-specific hybridization. Some neural-like computing models, see e.g. spiking neural networks [48]–[50] and parallel computing models [51] can be considered to design reliable DNA codes.

REFERENCES

[1] L. M. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, no. 5187, pp. 1021–1024, Nov. 1994.
[2] 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.
[3] Z. F. Qiu and M. Lu, "Take advantage of the computing power of DNA computers," in *International Parallel and Distributed Processing Symposium*. Berlin, Germany: Springer, 2000, pp. 570–577.
[4] P. W. K. Rothemund, "Folding DNA to create nanoscale shapes and patterns," *Nature*, vol. 440, no. 7082, pp. 297–302, Mar. 2006.
[5] 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.
[6] 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.
[7] 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.
[8] M. H. Garzon and R. J. Deaton, "Biomolecular Computing and Programming," *IEEE Trans. Evol. Comput.*, vol. 3, no. 3, pp. 236–250, Sep. 1999.
[9] R. Deaton, M. Garzon, R. C. Murphy, D. R. Franschetti, J. A. Rose, and S. E. Stevens, "Information transfer through hybridization reactions in DNA based computing," in *Proc. 2nd Annu. Conf.*, Jul. 1997, pp. 463–471.
[10] R. Deaton and M. Garzon, "Thermodynamic constraints on DNA-based computing," in *Proc. Comput. Bio-Molecules, Theory Exp.*, Sep. 1998, pp. 138–152.
[11] 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.
[12] A. J. Hartemink, D. K. Gifford, and J. Khodor, "Automated constraint-based nucleotide sequence selection for DNA computation," *Biosystems*, vol. 52, nos. 1–3, pp. 227–235, Oct. 1999.
[13] R. Penchovsky and J. Ackermann, "DNA library design for molecular computation," *J. Comput. Biology.*, vol. 10, no. 2, pp. 215–229, Jul. 2003.
[14] M. Arit and S. Kobayashi, "DNA sequence design using templates," *New Gener. Comput.*, vol. 20, no. 3, pp. 263–277, Sep. 2002.
[15] W. Liu, S. Wang, L. Gao, F. Zhang, and J. Xu, "DNA sequence design based on template strategy," *J. Chem. Inf. Comput. Sci.*, vol. 43, no. 6, pp. 2014–2018, Aug. 2003.
[16] U. Feldkamp, S. Saghafi, W. Banzhaf, and H. Rauhe, "DNASequence-Generator: A Program for the construction of DNA sequences," in *International Workshop on DNA-Based Computers*. Berlin, Germany: Springer, 2001, pp. 23–32.
[17] F. Tanaka, M. Nakatsugawa, M. Yamamoto, T. Shiba, and A. Ohuchi, "Towards a general-purpose sequence design system in DNA computing," in *Proc. Congr. Evol. Comput.(CEC)*, vol. 1, May 2002, pp. 73–78.
[18] Z. Ibrahim, N. K. Khalid, K. S. Lim, S. Buyamin, and J. A. A. Mukred, "A binary vector evaluated particle swarm optimization based method for DNA sequence design problem," in *Proc. IEEE Student Conf. Res. Develop.*, Dec. 2011, pp. 160–164.
[19] R. Deaton, M. Garzon, R. C. Murphy, J. A. Rose, D. R. Franceschetti, and S. E. Stevens, "Reliability and efficiency of a DNA-based computation," *Phys. Rev. Lett.*, vol. 80, no. 417, pp. 417–420, Jan. 1998.
[20] A. J. Ruben, S. J. Freeland, and L. F. Landweber, "PUNCH: An evolutionary algorithm for optimizing bit set selection," in *International Workshop on DNA-Based Computers*. Berlin, Germany: Springer, 2001, pp. 150–160.
[21] B. T. Zhang and S. Y. Shin, "Molecular algorithms for efficient and reliable DNA computing," in *Proc. Issues Supply Chain Scheduling Contracting*, 1998, pp. 735–742.
[22] M. Arita, A. Nishikawa, M. Hagiya, K. Komiya, H. Gouzu, and K. Sakamoto, "Improving sequence design for DNA computing," in *Proc. 2nd Annu. Conf. Genetic Evol. Comput.* Burlington, MA, USA: Morgan Kaufmann, Jul. 2000, pp. 875–882.
[23] S.-Y. Shin, D.-M. Kim, I.-H. Lee, and B.-T. Zhang, "Evolutionary sequence generation for reliable DNA computing," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 1, May 2002, pp. 79–84.
[24] S.-Y. Shin, I.-H. Lee, D. Kim, and B.-T. Zhang, "Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing," *IEEE Trans. Evol. Comput.*, vol. 9, no. 2, pp. 143–158, Apr. 2005.
[25] C. Xu, Q. Zhang, B. Wang, and R. Zhang, "Research on the DNA sequence design based on GA/PSO algorithms," in *Proc. 2nd Int. Conf. Bioinf. Biomed. Eng.*, May 2008, pp. 816–819.
[26] G. Cui and X. Li, "The optimization of DNA encodings based on modified PSO/GA algorithm," in *Proc. Int. Conf. Comput. Des. Appl.*, Jun. 2010, pp. 609–614.
[27] N. K. Khalid, T. B. Kurniawan, Z. Ibrahim, Z. M. Yusof, M. Khalid, and A. P. Engelbrecht, "A model to optimize DNA sequences based on particle swarm optimization," in *Proc. 2nd Asia Int. Conf. Modelling Simulation*, May 2008, pp. 534–539.
[28] T. B. Kurniawan, N. K. Khalid, Z. Ibrahim, M. S. Z. Abidin, and M. Khalid, "Sequence design for direct-proportional length-based DNA computing using population-based ant colony optimization," in *Proc. ICCAS-SICE*, Aug. 2009, pp. 1486–1491.
[29] T. B. Kurniawan, N. K. Khalid, Z. Ibrahim, M. Khalid, and M. Middendorf, "Evaluation of ordering methods for DNA sequence design based on ant colony system," in *Proc. 2nd Asia Int. Conf. Modelling Simulation*, May 2008, pp. 905–910.
[30] 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," *Math. Comput. Simul.*, vol. 151, pp. 131–139, Sep. 2018.
[31] X. S. Yang, "A new metaheuristic bat-inspired algorithm," *Comput. Knowl. Technol.*, vol. 284, pp. 65–74, 2010.
[32] V. M. Cervantes-Salido, O. Jaime, and I. M. Martínez-Pérez, "Improving the design of sequences for DNA computing: A multiobjective evolutionary approach," *Appl. Soft Comput.*, vol. 13, no. 12, pp. 4594–4607, Dec. 2013.
[33] J. Xiao, X. Zhang, and J. Xu, "A membrane evolutionary algorithm for DNA sequence design in DNA computing," *Sci. Bulletin.*, vol. 57, no. 6, pp. 698–706, Feb. 2012.

[34] J. G. Wetmur, "DNA probes: Applications of the principles of nucleic acid hybridization," *Critical Rev. Biochem. Mol. Biol.*, vol. 26, nos. 3–4, pp. 227–259, 1991.
[35] J. SantaLucia, "A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics," *Proc. Nat. Acad. Sci. USA*, vol. 95, no. 4, pp. 1460–1465, 1998.
[36] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Human Sci. (MHS)* Oct. 2002, pp. 39–43.
[37] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. ICNN-Int. Conf. Neural Netw.*, Perth, Australia, Nov./Dec. 1995, pp. 1942–1948.
[38] G. Yang, "A modified particle swarm optimizer algorithm," in *Proc. 8th Int. Conf. Electron. Meas. Instrum.*, Aug. 2007, pp. 675–679.
[39] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[40] M. T. Tommiska, "Efficient digital implementation of the sigmoid function for reprogrammable logic," *IEE Proc.-Comput. Digit. Techn.*, vol. 150, no. 6, pp. 403–411, Nov. 2003.
[41] P. W. Tsai, J. S. Pan, B. Y. Liao, M. J. Tsai, and V. Istanda, "Bat algorithm inspired algorithm for solving numerical optimization problems," *Appl. Mech. Mater.*, vols. 148–149, pp. 134–137, Dec. 2011.
[42] T. Niknam and B. Amiri, "An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis," *Appl. Soft Comput.*, vol. 10, no. 1, pp. 183–197, Jan. 2010.
[43] Y. Liu, X. Zheng, B. Wang, S. Zhou, and C. Zhou, "The optimization of DNA encoding based on chaotic optimization particle swarm algorithm," *J. Comput. Theor. Nanosci.*, vol. 13, no. 1, pp. 443–449, Jan. 2016.
[44] J. Xiao, J. Xu, Z. Chen, K Zhang, and L. Pan, "A hybrid quantum chaotic swarm evolutionary algorithm for DNA encoding," *Comput. Math. Appl.*, vol. 57, nos. 11–12, pp. 1949–1958, Jun. 2009.
[45] D. F. Luo and D. J. Luo, "The research of DNA coding sequences based on invasive weed optimization," *Sci. Technol. Eng.*, vol. 13, pp. 3545–3551, 2013,.
[46] L. Huang, I. H. Suh, and A. Abraham, "Dynamic multi-objective optimization based on membrane computing for control of time-varying unstable plants," *Inf. Sci.*, vol. 181, no. 11, pp. 2370–2391, Jun. 2011.
[47] J.-H. Xiao, Y. Jiang, J.-J. He, and Z. Cheng, "A dynamic membrane evolutionary algorithm for solving dna sequences design with minimum free energy," *Match Commun. Math. Comput. Chem.*, vol. 70, no. 3, pp. 987–1004, 2013.
[48] T. Song, X. Zeng, P. Zheng, M. Jiang, and A. Rodríguez-Patón, "A parallel workflow pattern modeling using spiking neural P systems with colored spikes," *IEEE Trans. Nanobiosci.*, vol. 17, no. 4, pp. 474–484, Oct. 2018. doi: 10.1109/TNB.2018.2873221.
[49] T. Song, S. Pang, S. Hao, A. Rodríguez-Patón, and P. Zheng, "A parallel image skeletonizing method using spiking neural P systems with weights," in *Proc. Neural Process. Lett.*, Oct. 2018, pp. 1–18. doi: 10.1007/s11063-018-9947-9.
[50] T. Song, A. Rodríguez-Patón, P. Zheng, and X. Zeng, "Spiking neural P systems with colored spikes," *IEEE Trans. Cogn. Devel. Syst.*, vol. 10, no. 4, pp. 1106–1115, Dec. 2018. doi: 10.1109/TCDS.2017.2785332.
[51] T. Song, P. Zheng, M. L. D. Wong, and X. Wang, "Design of logic gates using spiking neural P systems with homogeneous neurons and astrocytes-like control," *Inf. Sci.*, vol. 372, pp. 380–391, Dec. 2016.