Applied Intelligence
https://doi.org/10.1007/s10489-022-03491-7

Hybrid genetic algorithms for the determination of DNA motifs to satisfy postulate 2-Optimality

Dai Tho Dang[1] · **Ngoc Thanh Nguyen**[2,3] · **Dosam Hwang**[4]

Accepted: 9 March 2022

Abstract
Currently, determining DNA motifs or consensus plays an indispensable role in bioinformatics. Many postulates have been proposed for finding a consensus. Postulate 2-Optimality is essential for this task. A consensus satisfying postulate 2-Optimality is the best representative of a profile, and its distances to the profile members are uniform. However, this postulate has not been widely investigated in identifying a DNA motif or consensus for a DNA motif profile. The HDC algorithm is the best at this task in the literature. This study focuses on determining DNA motifs that satisfy postulate 2-Optimality. We propose a new hybrid genetic (HG1) algorithm based on the elitism strategy and local search. Subsequently, a novel elitism strategy and longest distance strategy are introduced to maintain the balance of exploration and exploitation. A new hybrid genetic (HG2) algorithm is developed based on the proposed exploration and exploitation balance approach. The simulation results show that these algorithms provide a high-quality DNA motif. The HG2 algorithm provides a DNA motif with the best quality.

Keywords Evolutionary computation · Hybrid genetic algorithm · Postulate 2-Optimality · DNA motif · DNA sequence

1 Introduction

Determining consensus for sets of sequences is indispensable for bioinformatics [1]. A DNA motif is a nucleic acid sequence pattern with some biological functions. It often repeats in many genes or many times in one gene. DNA motifs are often associated with structural motifs found in proteins, which can occur on both strands of DNA [2]. Note that motifs are never precisely the same in terms of the actual conserved sequence. Various sequence variabilities exist with regard to a single motif [3].

Determining DNA motifs often consists of two stages. The first stage is the identification of a set of motif candidates from DNA sequences. In the second stage, the DNA motif is determined based on the identified motif candidates [4]. These results are shown in Fig. 1. In this study, a set of DNA motif candidates is called a DNA motif profile. Figure 2 is an example of a DNA motif profile from E. coli.

Many postulates are used to find consensus for a profile. Among these, 1-Optimality and 2-Optimality are the most important postulates. A consensus satisfying postulate 1-Optimality is named the 1-Optimality consensus, and a consensus satisfying postulate 2-Optimality is named the 2-Optimality consensus. However, finding a 1-Optimality or 2-Optimality consensus is an NP-hard problem [5]. In [5, 6], Nguyen proved that:

– The 1-Optimality consensus is the best representative of a profile.

This article belongs to the Topical Collection: *Emerging Topics in Artificial Intelligence Selected from IEA/AIE2021*
Guest Editors: Ali Selamat and Jerry Chun-Wei Lin

✉ Ngoc Thanh Nguyen
Ngoc-Thanh.Nguyen@pwr.edu.pl

✉ Dosam Hwang
dosamhwang@gmail.com

Dai Tho Dang
ddtho@vku.udn.vn; daithodang@ynu.ac.kr

[1] Vietnam - Korea University of Information and Communication Technology, The University of Danang, Danang, Vietnam

[2] Department of Applied Informatics, Faculty of Information and Communication Technology, Wrocław University of Science and Technology, Wrocław, Poland

[3] Faculty of Information Technology, Nguyen Tat Thanh University, Nguyen Tat Thanh, Vietnam

[4] Department of Computer Engineering, Yeungnam University, Yeungnam, Republic of Korea

Published online: 20 April 2022

Fig. 1 Stages of finding a DNA motif

– The 2-Optimality consensus is the best representative of the profile. In addition, its distances to the profile members are uniform.

Determining a consensus that satisfies postulates 2-Optimality has not been widely investigated for DNA structures. In our earlier work [7], we formulated the problem of finding a 2-Optimality consensus for DNA structures and proposed algorithms to solve this task. The HDC algorithm generates the highest consensus quality.

A genetic algorithm (GA) is a well-known evolutionary algorithm. It was first introduced by Holland [9] and is based on Charles Darwin's natural selection theory and genetics. The GA has been improved and developed to solve optimization problems [10]. Currently, the GA is a robust general-purpose optimizer that works surprisingly well in many applications [11, 12], especially in solving large complex scale optimization problems in many areas, such as economics [13], cybernetics [14], and bioinformatics [15].

Elitism is an effective strategy for enhancing the efficiency of evolutionary algorithms. This guarantees that the best solutions in each generation are saved for the next generation without any changes [16]. The solution quality obtained by the genetic algorithm does not decrease from one generation to the next, and the quality of each generation's best solution monotonically increases over time [17].

Exploration and exploitation are the two foundations that characterize the capabilities of evolutionary algorithms [18]. Exploration is the process of visiting a new region in a search space, while exploitation is the process of searching in the neighborhood of the recently visited regions. Maintaining a balance between exploration and exploitation is the key to the success of these algorithms [19]. For a GA, a type of evolutionary algorithm, Herrera and Lozano emphasized that the exploitation and exploration relationship determines the GA behavior maintained throughout the run [20].

Thus, this study extends the previous study [7] by using the elitism strategy and maintaining a balance between exploration and exploitation. The main contributions of this study are as follows:

– A new hybrid GA was developed based on an elitism strategy and local search (HG1).
– We proposed a novel elitism strategy and the longest distance strategy. A new approach for balancing exploration and exploration was developed, based on the novel elitism strategy and the longest distance strategy.
– A new hybrid GA was developed based on the new approach for maintaining the balance between exploration and exploitation (HG2).

The remainder of this paper is organized as follows: Some related works of this study are introduced in Section 2, and the problem formulation is presented in Section 3. Section 4 introduces the proposed algorithms, and Section 5 presents the simulations and evaluations. Some discussions are presented in Section 6. Finally, the conclusions are presented in Section 7.

2 Related works

Consensus problems have been investigated in many areas, such as bioinformatics [1], computer science [21, 22], and medicine [23]. Many postulates are used for consensus choice functions, i.e., the consistency, unanimity, condorcet consistency, general consistency, simplification, proportion, quasi-unanimity, reliability, 1-Optimality, and 2-Optimality postulates [5]. However, no consensus satisfies all the postulates simultaneously. The 1-Optimality and 2-Optimality postulates play an important role in determining consensus. The reason is that if one consensus satisfies each of them, it will satisfy most of the other postulates as well [5][6]. Postulate 1-Optimality requires a consensus to be as

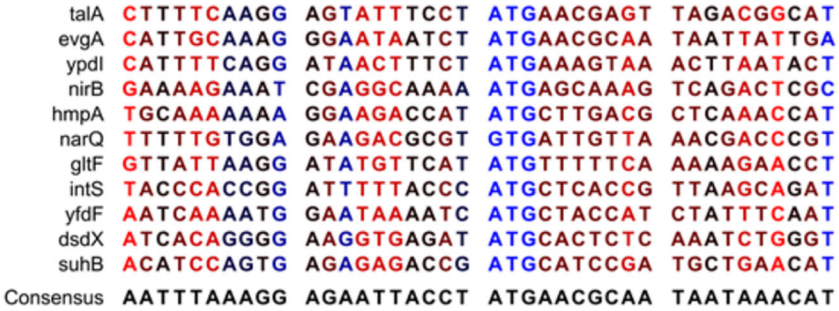

Fig. 2 Set of motif candidates and their DNA motifs from E. coli [8]

close as possible to the members of the profile. Postulate 2-Optimality requires the sum of the squared distances from a consensus to the profile members to be minimal [5, 6].

By $\prod(U)$, we denote a set of all nonempty finite subsets with repetitions of U. For a profile $S \in \prod(U)$, the consensus of S is determined by

- Postulate 1-Optimality if

$$d(c^*, S) = \min_{x \in U} d(x, S)$$

- Postulate 2-Optimality if

$$d^2(c^*, S) = \min_{x \in U} d^2(x, S)$$

where c^* is a consensus of profile S, d is the sum of the distances between c^* and the profile members, and d^2 is the sum of the squared distances between c^* and the profile members. Finding a 1-Optimality consensus or a 2-Optimality consensus is an NP-hard problem [5]. For the 2-Optimality consensus, heuristic algorithms are used for many data structures, such as binary vectors [24], proteins [25], and phylogenetic trees [26].

Finding a DNA motif or consensus for a DNA motif profile has been investigated for a long time. Two widespread conditions are minimizing the longest distance from the profile elements to the consensus and minimizing the sum of distances from the profile elements to the consensus [27].

There are three approaches for determining a DNA motif in the literature: probabilistic approaches, approaches based on enumeration, and evolutionary approaches [28]. In the probabilistic approach, the expectation-maximization algorithm is a good example. This algorithm includes two stages: expectation and maximization. The expectation stage estimates the values of unknowns based on parameters, and the maximization stage utilizes these estimates to improve the parameters across several repetitions [29]. In addition, numerous algorithms have been proposed based on the Bayesian model, Bayesian Markov model, and hidden Markov model [28]. The enumeration approach predicts motifs based on the enumeration of words and word similarities [28]. The time complexity of algorithms based on this approach is exponential, and they can solve for motifs with lengths of six or shorter. DREME, CisFinder, Weeder, and FMotif are well-known algorithms [28, 30]. Most algorithms developed based on the enumeration or probabilistic approaches are time-consuming and become easily trapped in local optima [28]. The evolutionary approach is expected to yield good results for DNA detection. The GA-DPAF algorithm for finding dyad patterns is a standard GA that employs a multiobjective fitness function and crossover and mutation operations [31]. Wei et al. presented a GA for motif elicitation in [32]. The ADJUST and SHIFT operators were used to overcome the local optima. Another evolutionary algorithm was introduced in [33]. The new population is formed by holding the best parents and using crossover and mutation operators.

Genetic algorithms (GAs) are widely used to solve optimization problems [10]. To be successful, GAs need to balance exploration and exploitation [19]. Many studies have focused on solving this problem using GAs. There are three main approaches for balancing exploitation and exploration in the literature [18, 19].

The diversity maintenance approach assumes that "*the proposed techniques will maintain diversity per se, and hence the balance between exploration and exploitation will be achieved*" [19]. The saw-tooth GA uses a variable population size and cyclic partial reinitialization in a saw-tooth function to obtain better population diversity and performance [34]. In [35], the authors introduced a survival strategy that replaced elements with weak fitness and low diversity. This strategy increases population diversity and solution quality. Hutter et al. maintained genetic diversity through fitness diversity [36]. Fitness values were classified into many classes, and each class had an equal opportunity to survive. Thus, diversity was preserved through selection.

The diversity control approach measures population diversity and uses it as feedback to choose exploitation or exploration [19]. In [37], the authors introduced an adaptive genetic operator to choose a suitable number of high-fitness individuals. This operator uses the number of outliers to set the ratio of the exploitation and exploration phases. An adaptive GA was proposed in [38]. The adaptive probabilities of mutation and crossover are applied to manipulate population diversity and achieve good convergence.

The diversity learning approach [19] uses long-term history and machine learning to study unexplored search areas. In GASOM [39], self-organizing maps are used to mine data from the evolution process. The algorithm can achieve population diversity and a balance between exploitation and exploration because the population has individuals with high novelty. In the nonrevisiting GA [40], previous solutions are never revisited, and population diversity is automatically warranted.

Only one study has investigated the 2-Optimality consensus for DNA motif profiles [7]. The study introduced three GAs for this task, in which the HDC algorithm is the best. This algorithm is developed by combining the traditional GA and a local search algorithm. However, the problem of maintaining a balance between exploration and exploitation was not considered.

3 Problem formulation

A string is generally understood as a sequence of characters from a finite alphabet Σ. A DNA sequence is described as a string of characters (or symbols) from Σ={A, C, T, G}. Each

letter represents a standard amino acid adenine (A), cytosine (C), guanine (G), and thymine (T). In general, string and sequence terms are often used equivalently.

Definition 1 The length of a DNA string s is the number of symbols it comprises.

We denote the length of s_i as $|s_i|$. For example, s_1="GAAGCCGGATT" is a DNA string of length 11, and s_2= "AGCTGGACG" is a DNA string of length 9. Let U denote a finite set of all strings over Σ with length m.

Definition 2 For $|s_1| = |s_2| = m, s_1 = s_1^1, s_1^2, ..., s_1^m, s_2 = s_2^1, s_2^2, ..., s_2^m$; $s_1, s_2 \in U$, the distance between s_1 and s_2 is calculated as follows:

$$d(s_1, s_2) = \sum_{i=1}^{m} \sigma(s_1^i, s_2^i)$$

where

$$\sigma(s_1^i, s_2^i) = \begin{cases} 0, & \text{if } s_1^i = s_2^i \\ 1, & \text{if } s_1^i \neq s_2^i \end{cases}$$

for $1 \leq i \leq m$.

In this study, a set of DNA motifs is considered a DNA motif profile.

Definition 3 A DNA motif profile with size n is defined as

$$S = \{s_1, s_2, ..., s_n\}$$

where $s_i \in U$ and $|s_i| = m$ for $1 \leq i \leq n$.

Definition 4 For a DNA motif profile S, the consensus of S is determined by postulate 2-Optimality if

$$d^2(c^*, S) = \min_{x \in U} d^2(x, S)$$

where c^* is a 2-Optimality consensus of S.

The problem of determining the 2-Optimality consensus for a DNA motif profile is a discrete optimization problem. Such problems involve searching for the best solution in an exponentially large set of solutions [7]. Genetic algorithms have proven to be efficient tools for solving optimization problems [10, 11].

Definition 5 The fitness function for determining the 2-Optimality consensus for a DNA motif profile is

$$f(x) = \min_{x \in U} d^2(x, S)$$

Once the population is initialized or an offspring population is generated, the fitness value of each candidate solution is evaluated. The determination of the fitness function plays an important role in GAs. This function takes an individual (or solution) and measures how "fit" the individual is for the problem [9, 10].

Consider a profile

$$S = \{s_1, s_2, s_3\}$$

where s_1="ACCATCC", s_2="ATCATAG", and s_3="ACCATGA". For x= "ACCAACA", the distances from x to s_1, s_2, and s_3 are 2, 4, and 2, respectively. The sum of the squared distances from x to members of S is 24. In other words, we have $f(x)$=24.

4 Proposed methods

4.1 HG1 algorithm

Elitism strategies have long been regarded as an effective method for enhancing the efficiency of GAs [41]. This ensures that elitist individuals are always propagated to the next generation. Without an elitism strategy, the best individual can be lost because of stochastic errors [16].

In the elitism strategy, the best one or more individuals in each generation are inserted into the next generation without undergoing any change. Using the elitism strategy, the best solutions are never lost during the search process. Thus, the quality of the best solution in each generation increases monotonically over time [17]. We propose an algorithm that combines a GA with a new elitism strategy and local search to find the 2-Optimality consensus for a DNA motif profile.

The HG1 algorithm begins with initializing individuals and created population P of a predefined size N_P. The number of elites, E, is N_E. Elite E is passed to the next generation. Population P goes through selection, crossover, and mutation to generate offspring O of size N_O. Next, P without the elements in E and offspring O are merged to the solution set M of size $N_P + N_O - N_E$. The survival reduces M to a solution set of size N_P by selecting $N_P - N_E$ fittest individuals and the N_E elites. Then, the new population is applied to the next generation for recombination. This process terminates if it reaches a given number of generations G_{max} ($gen = G_{max}$).

In practice, the number of elites is often set to a small value, i.e., $N_E = 1$ or $N_E = 2$. This algorithm sets $N_E = 2$.

The schema of the HG1 algorithm is shown in Fig. 3.

Initialization: Random initialization creates the initial population with random solutions, and heuristic initialization generates the population utilizing a heuristic for the problem. The population generated by heuristic initialization often has similar solutions and low diversity, but that generated by random initialization has a solution that can progress the population toward optimality [11, 16, 19]. Thus, the

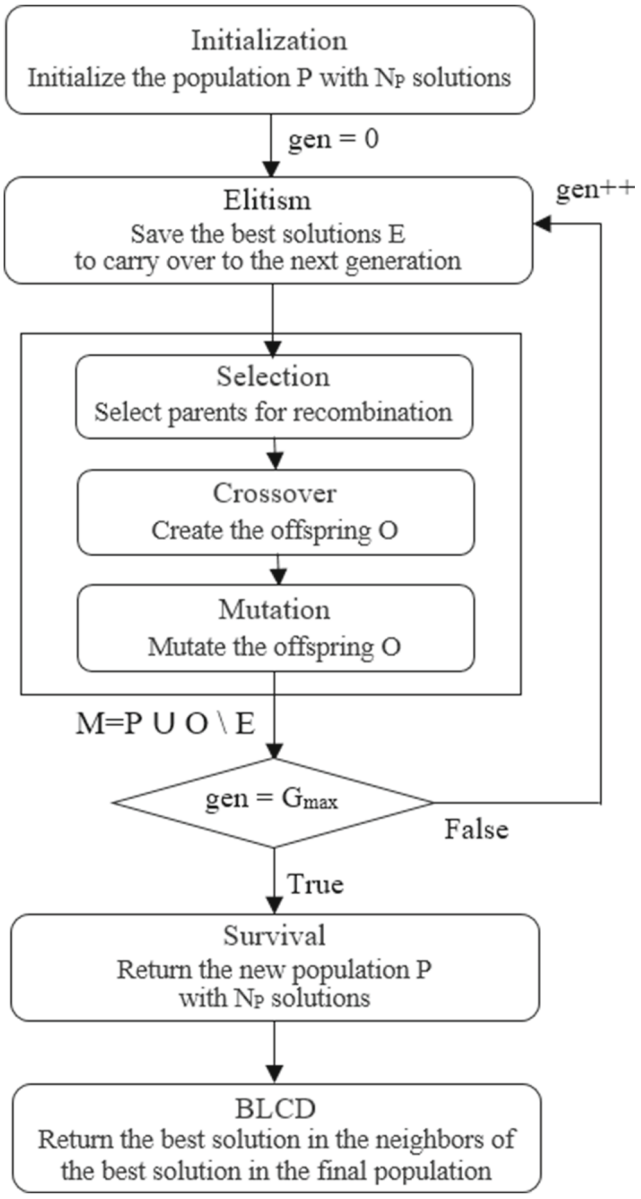

Fig. 3 Schema of the HG1 algorithm

algorithm randomly generates the initial population. Each individual is a string over Σ={A, C, T, G}, and the length of the string is m.

Elitism: The elitism strategy is analyzed above. The two best solutions are chosen for insertion into the next generation.

Selection (selection of mating pairs): This process selects parents who mate and recombine to generate offspring for the new generation. Tournament selection is considered the most popular selection technique because its effectiveness has been widely demonstrated. The standard tournament selects k individuals randomly from the population and chooses the best among them to become a parent [42, 43]. This process is repeated to choose the next parent. The study uses tournament selection with a size of three (k=3).

Crossover: Crossover recombines the chromosomes to create two offspring from two parents. This directly influences the search process for obtaining good solutions. Recombination of the good genes of one chromosome with the good genes of another should generate a better chromosome [9, 16]. This study utilizes a two-point operator. First, two points are randomly chosen from the parent chromosomes. Next, it generates two offspring by swapping the genes between the two points.

Mutation: The use of crossover operators can provide better solutions. In the case where two parents have the same allele at the same gene, after performing the crossover operator, the allele of the gene is not changed. In the worst case, the population has the same allele at the same gene, and hence, the population has the same allele permanently. Mutations bring new characteristics into the population [9, 38]. We used a triple-value operator that randomly chooses an individual and alters one individual gene with the other three bases [7]. Figure 4 shows an example of the tripe-value operator.

Survival: Set M is formed from population P without elites E and offspring O:

$$M = P \cup O \setminus E$$

The size of set M is $N_P + N_O - N_E$. The next population is formed from $N_P - 2$ fittest individuals who survive from M and the two elites.

BLCD: BLCD is a local search algorithm that was proposed in a previous study [7]. It finds the best solution among the neighbors of the best individual in the final population. The neighbors b_i of the solution e_i differ from e_i by one gene. The result of this algorithm is the output of the HG1 algorithm.

4.2 HG2 algorithm

Exploration and exploitation are the two cornerstones of problem-solving by searching [18]. Maintaining a balance between exploration and exploitation is key to the success of GAs. Hence, a balance between exploitation and exploration necessitates direct control. However, researchers have not found a way to measure the balance between exploitation and exploration directly, and they rely on indirect measures of exploitation and exploration, especially population diversity [19, 44].

The elitism strategy always keeps the most befitting candidates and is helpful for exploitation. The strategy is widely used because of its significant advantages. It finds

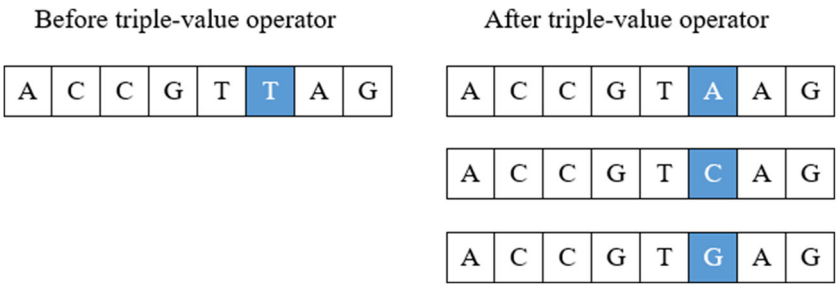

Fig. 4 Tripe-value operator

and passes elites to the next generation [44]. This study proposes a novel elitism strategy called IEBL. The IEBL strategy consists of two steps.

- First, it finds set E with N_E elites.
- Second, it increases the quality of elites in E and gives the set EB.

We can observe that the previous elitism strategies only perform the first step and they do not improve the quality of the elites. Let $E = \{e_1, e_2, ..., e_{N_E}\}$ be a set of elites of the population. For each elite e_i, we use a local search to find the best solution b_i in the neighbor of elitist e_i. Let $EB = \{b_1, b_2, ..., b_{N_E}\}$ be a set of b_i. It is clear that each element in the set EB is better or equal to the corresponding elements in set E. The set EB is passed to the next generation instead of set E. Our elitism strategy is as follows.

However, although an elitism strategy can always significantly improve convergence speed, it does not always benefit the balance between exploration and exploitation. The reason for this situation is that it emphasizes exploitation. A measure to enhance exploration is required to cooperate with the elitism strategy [19].

Elitism usually speeds up the convergence of the algorithm and reduces population diversity. When population diversity is too low, crossover rarely generates offspring that are different from parents. Thus, there is a low possibility of obtaining healthier offspring than the parents [19, 44]. In many circumstances, diversity needs to be maintained beyond a certain level. If elites are always kept in the population under the control of an elitism strategy, simultaneously maintaining individuals that are far from the elites in the population is a possible approach to maintaining diversity without making excess degeneracy [44].

Algorithm 1 IEBL;

Input: $E = \{e_1, e_2, ..., e_{N_E}\}$;
Output: $EB = \{b_1, b_2, ..., b_{N_E}\}$;
1: EB={};
2: **for** each e_i in E **do**
3: Search for the best solution b_i in neighbors of elitist e_i;
4: $EB = EB \cup \{b_i\}$;
5: **end for**
6: **return** EB;

Our strategy that keeps individuals that are far from elites is called the KLD strategy. Set EB is saved for the next generation. For each member b_i in the set EB, we look for l_i

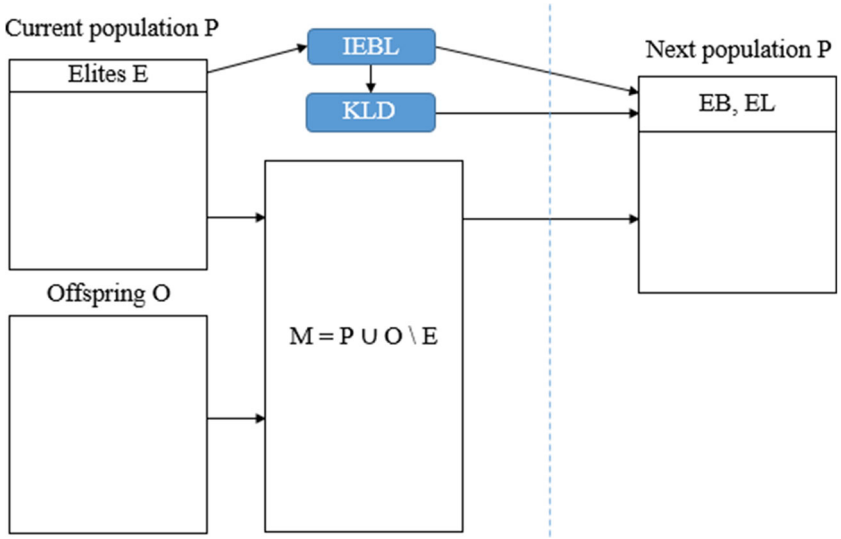

Fig. 5 Balance of exploration and exploitation

in M that is the farthest distance from b_i. Finally, we obtain the set EL that is formed by all the longest-distance elements l_i. The set EL is saved for the next generation.

Algorithm 2 KLD;

Input: $EB = \{b_1, b_2, ..., b_{N_E}\}$; $M = P \cup O \setminus E$
Output: $EL = \{l_1, l_2, ..., l_{N_E}\}$;
1: EL={};
2: **for** each b_i in EB **do**
3: Search for l_i in M having the longest distance to b_i;
4: $EL = EL \cup \{l_i\}$;
5: **end for**
6: **return** EL;

The novel elitism strategy and longest distance strategy maintain the balance between exploration and exploitation, as shown in Fig. 5.

The next population is formed by:

- N_E elements from EB;
- N_E elements from EL;
- $N_P - 2N_E$ elements from the fittest individuals.

The time complexity of the KLD algorithm is $O(N_E m^2 n)$. In practice, the number of elites is often set to one or two. This algorithm sets $N_E = 1$ and its complexity is $O(m^2 n)$.

A schema of the HG2 algorithm is shown in Fig. 6. In this algorithm, the initialization, evaluation, selection, crossover, mutation, survival, stopping criterion, and BLCD are the same as those in the HG1 algorithm. The IEBL and KLD algorithms are introduced above.

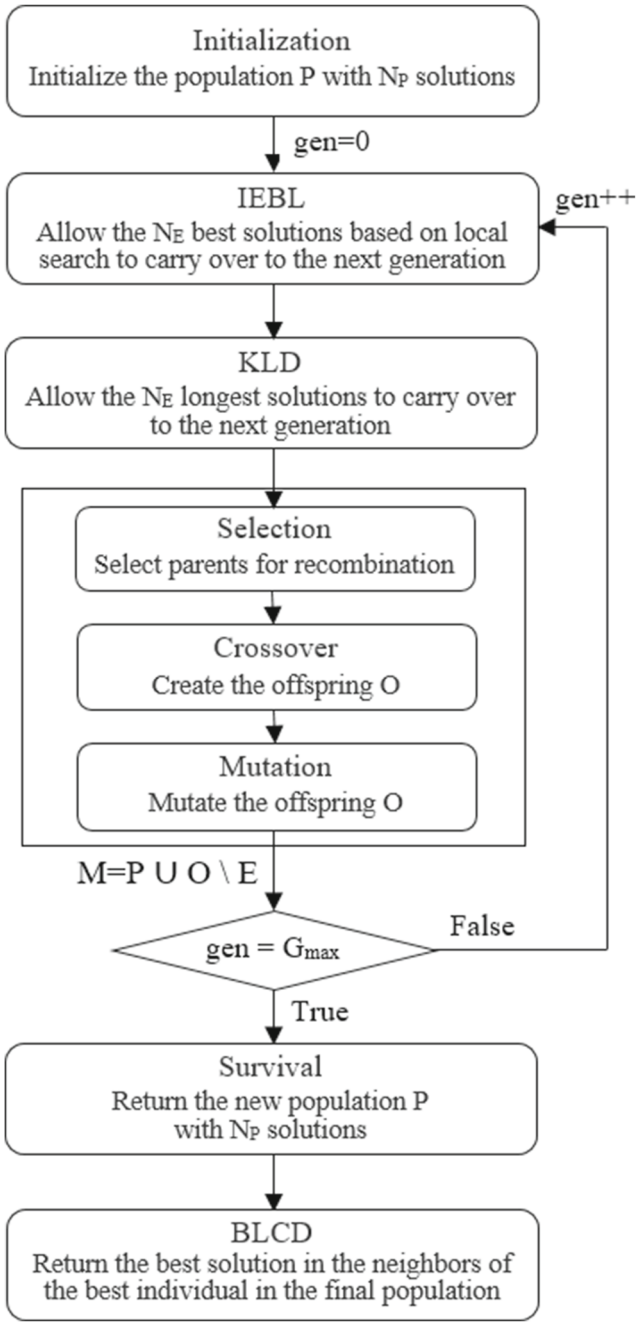

Fig. 6 Schema of the HG2 algorithm

5 Simulations and evaluation

This section describes the simulations performed to evaluate the HG1 and HG2 algorithms. Evaluations are made by comparing the HG1 and HG2 algorithms with the HDC algorithm in terms of execution time and consensus quality. We used the HDC algorithm because it was the best algorithm in a previous study [7].

5.1 Consensus Quality

The consensus quality is calculated as follows:

$$Qu = \frac{|d^2(c, S) - d^2(c^*, S)|}{d^2(c^*, S)}$$

where c^* is the optimal consensus and c is a consensus generated by the HDC, HG1, or HG2 algorithm. The optimal consensus c^* is generated by the brute-force algorithm.

The dataset or the DNA motif profile was generated from https://www.bioinformatics.org/sms2/. The size of the DNA motif profile was 500, and the length of the elements (or motifs) was 20.

The parameters of the three algorithms are set as follows:

- The population size is 30.
- The number of generations is 25.
- The number of paired parents is 30.
- The number of offspring is 90.

Table 1 Consensus quality of the algorithms

ORD	HDC	HG1	HG2	ORD	HDC	HG1	HG2	ORD	HDC	HG1	HG2
1	1.0000	1.0000	1.0000	41	1.0000	1.0000	1.0000	81	1.0000	1.0000	1.0000
2	1.0000	0.9998	1.0000	42	1.0000	1.0000	1.0000	82	0.9980	1.0000	1.0000
3	1.0000	1.0000	1.0000	43	1.0000	1.0000	1.0000	83	1.0000	1.0000	1.0000
4	1.0000	1.0000	1.0000	44	1.0000	1.0000	1.0000	84	0.9978	1.0000	1.0000
5	1.0000	1.0000	1.0000	45	1.0000	1.0000	1.0000	85	1.0000	0.9988	1.0000
6	1.0000	1.0000	1.0000	46	1.0000	1.0000	1.0000	86	1.0000	1.0000	1.0000
7	1.0000	1.0000	1.0000	47	0.9987	0.9989	1.0000	87	1.0000	1.0000	1.0000
8	1.0000	1.0000	1.0000	48	1.0000	1.0000	1.0000	88	1.0000	1.0000	1.0000
9	1.0000	1.0000	1.0000	49	1.0000	1.0000	1.0000	89	1.0000	1.0000	1.0000
10	1.0000	1.0000	1.0000	50	1.0000	1.0000	1.0000	90	1.0000	1.0000	1.0000
11	0.9981	1.0000	1.0000	51	1.0000	1.0000	1.0000	91	1.0000	1.0000	1.0000
12	1.0000	1.0000	1.0000	52	0.9980	1.0000	1.0000	92	1.0000	1.0000	1.0000
13	1.0000	1.0000	1.0000	53	1.0000	1.0000	1.0000	93	1.0000	1.0000	1.0000
14	1.0000	1.0000	1.0000	54	1.0000	1.0000	1.0000	94	1.0000	1.0000	1.0000
15	1.0000	0.9981	1.0000	55	1.0000	1.0000	1.0000	95	1.0000	1.0000	1.0000
16	1.0000	1.0000	1.0000	56	1.0000	1.0000	1.0000	96	1.0000	1.0000	1.0000
17	1.0000	1.0000	1.0000	57	1.0000	1.0000	1.0000	97	1.0000	0.9982	1.0000
18	1.0000	1.0000	1.0000	58	1.0000	1.0000	1.0000	98	1.0000	1.0000	1.0000
19	1.0000	1.0000	1.0000	59	1.0000	0.9986	1.0000	99	1.0000	1.0000	1.0000
20	1.0000	1.0000	1.0000	60	1.0000	1.0000	1.0000	100	1.0000	1.0000	1.0000
21	1.0000	1.0000	1.0000	61	1.0000	1.0000	1.0000	101	1.0000	0.9980	1.0000
22	1.0000	0.9983	1.0000	62	1.0000	1.0000	1.0000	102	1.0000	1.0000	1.0000
23	1.0000	1.0000	1.0000	63	1.0000	1.0000	1.0000	103	1.0000	1.0000	1.0000
24	1.0000	1.0000	1.0000	64	1.0000	1.0000	1.0000	104	1.0000	1.0000	1.0000
25	0.9980	1.0000	1.0000	65	1.0000	1.0000	1.0000	105	1.0000	1.0000	1.0000
26	1.0000	1.0000	1.0000	66	1.0000	1.0000	1.0000	106	1.0000	1.0000	1.0000
27	1.0000	1.0000	1.0000	67	1.0000	1.0000	1.0000	107	1.0000	1.0000	1.0000
28	1.0000	1.0000	1.0000	68	1.0000	1.0000	1.0000	108	1.0000	1.0000	1.0000
29	1.0000	1.0000	1.0000	69	1.0000	1.0000	1.0000	109	1.0000	1.0000	1.0000
30	1.0000	1.0000	1.0000	70	1.0000	1.0000	1.0000	110	1.0000	1.0000	1.0000
31	1.0000	1.0000	1.0000	71	1.0000	1.0000	1.0000	111	1.0000	0.9975	1.0000
32	1.0000	1.0000	1.0000	72	1.0000	1.0000	1.0000	112	1.0000	1.0000	1.0000
33	1.0000	1.0000	1.0000	73	1.0000	1.0000	1.0000	113	1.0000	1.0000	1.0000
34	1.0000	1.0000	1.0000	74	1.0000	1.0000	1.0000	114	1.0000	1.0000	1.0000
35	1.0000	1.0000	1.0000	75	1.0000	1.0000	1.0000	115	1.0000	1.0000	1.0000
36	1.0000	1.0000	1.0000	76	1.0000	1.0000	1.0000	116	0.9984	1.0000	1.0000
37	1.0000	1.0000	1.0000	77	1.0000	1.0000	1.0000	117	1.0000	1.0000	1.0000
38	1.0000	0.9988	1.0000	78	0.9991	1.0000	1.0000	118	1.0000	1.0000	1.0000
39	1.0000	1.0000	1.0000	79	1.0000	1.0000	1.0000	119	1.0000	1.0000	1.0000
40	1.0000	1.0000	1.0000	80	1.0000	0.9985	1.0000	120	1.0000	1.0000	1.0000

– The mutation rate is 0.01.

Each algorithm was executed on the dataset for 120 iterations; at each iteration, we received a consensus and its quality. Finally, we obtained three consensus quality samples.

– A consensus quality sample of the HDC algorithm.
– A consensus quality sample of the HG1 algorithm.
– A consensus quality sample of the HG2 algorithm.

The results are presented in Table 1. A significance level α of 0.05 was selected. The Shapiro–Wilk test was used to test whether the consensus quality samples of the three algorithms came from a normal distribution. The *p-values* of the consensus quality sample generated by the HDC, HG1, and HG2 algorithms were 2.2e-16, 2.1e-16, and 2.2e-16, respectively. Their *p-values* were less than the significance level. Thus, there is evidence that the consensus quality samples of these algorithms are not normally distributed.

We compared the consensus quality of the algorithms. The hypotheses for this test were as follows:

– H_0: The consensus qualities of these three algorithms are equal.
– H_1: The consensus qualities of the three algorithms are not equal.

The consensus quality samples of these algorithms were not normally distributed. The Kruskal–Wallis test was used for this. We obtained a *p-value* of 0.039. Thus, hypothesis H_0 can be rejected. In other words, the consensus qualities of the three algorithms were not equal.

The medians of the consensus-quality samples were compared in pairs. The consensus quality of the HG2 algorithm was 0.013% and 0.01% higher than those of the HG1 and HDC algorithms. The consensus quality of the HDC algorithm was 0.002% higher than that of HG1.

5.2 Running time

The dataset consisted of 10 DNA motif profiles. The length of each element (or motif) was 20. The profile sizes were 300, 350, 400, 450, 500, 550, 600, 650, 700, and 750. Three algorithms were applied to the dataset. We obtained three running time samples, one for each of the HDC, HG1, and HG2 algorithms. The results are presented in Table 2.

A significance level α of 0.05 was selected. The Shapiro-Wilk test was used to test whether the running time samples of the three algorithms come from normal distributions. The *p-values* of the consensus quality sample of the HDC, HG1, and HG2 algorithms were 0.75, 0.71, and 0.68, respectively.

Table 2 Running times of the algorithms (seconds)

Profile size	HDC algorithm	HG1 algorithm	HG2 algorithm
300	0.210	0.227	0.259
350	0.242	0.251	0.306
400	0.277	0.292	0.341
450	0.324	0.339	0.499
500	0.399	0.428	0.490
550	0.437	0.454	0.503
600	0.495	0.510	0.568
650	0.536	0.578	0.609
700	0.591	0.621	0.668
750	0.642	0.697	0.725

Thus, there is evidence that the running time samples of these algorithms are normally distributed.

We compared the running times of these algorithms. The hypotheses for this test were as follows:

- H_0: The running times of the three algorithms are equal.
- H_1: The running times of the three algorithms are not equal.

Samples were obtained from normal distributions; thus, the one-way ANOVA test was chosen for this investigation. We obtained a *p-value* of 0.02. Therefore, H_0 was rejected. In other words, the running times of the algorithms were not equal.

The variances of the samples were unequal, and Tamhane's T2 post hoc test was used to compare the running time of these algorithms. The output showed that the running time samples of the HDC and HG1 algorithms were 83.60% and 88.51% of the HG2 algorithm, respectively.

6 Discussion

The consensus quality of the HG1 algorithm was lower than that of the HDC algorithm. This was because the HG1 algorithm used the traditional elitism strategy. The elitism strategy accelerated the convergence of the algorithm. However, it also quickly reduced the population diversity. If the population diversity is too low, crossover seldom produces offspring that are different from the parents. This approach can get stuck in local optima.

The new approach for balancing exploration and exploration was developed based on the novel elitism strategy and the longest distance strategy. *EB* was better than or equal to *E*. Thus, the elites of the HG2 algorithm were better than or equal to those of the HG1 algorithm. The HG2 algorithm keeps the most befitting candidates and is helpful for exploitation. In addition, this algorithm kept individuals far from the elites. Thus, it could maintain diversity. This could be why the quality of consensus or motifs generated by the HG2 algorithm was the highest. In this study, the consensus quality of HG2 was always 1. This means that HG2 always generates the optimal consensus.

7 Conclusions

This study proposed two hybrid genetic algorithms to determine the 2-Optimality consensus for a DNA motif profile. The HG1 algorithm was based on an elitism strategy and local search. To maintain the balance between exploration and exploitation, we proposed a novel elitism strategy and the longest distance strategy. Based on this balance, the HG2 algorithm was developed. The simulation results showed that the consensus quality of the HG2 algorithm was 0.013% and 0.01% higher than those of the HG1 and HDC algorithms, respectively. The running time samples of the HDC and HG1 algorithms were 83.60% and 88.51% of the HG2 algorithm, respectively.

In the future, we will focus on finding a set of motif candidates and complete the problem of determining a DNA motif for DNA sequences. In addition, we will investigate motifs of other structures, such as protein motifs.

References

1. Eren K, Murrell B (2018) RIFRAF: A Frame-resolving Consensus Algorithm. Bioinformatics 34(22):3817–3824
2. Popa O, Oldenburg E, Ebenhöh O (2020) From sequence to information. Phil Trans R Soc B 375:20190448. https://doi.org/10.1098/rstb.2019.0448
3. Pradhan M (2008) Motif discovery in biological sequences. San Jose State University, San Jose
4. Compeau P, Pevzner P (2015) Bioinformatics algorithms: an active learning approach. Active Learning Publisher, USA
5. Nguyen NT (2008) Advanced methods for inconsistent knowledge management. Springer London, London
6. Nguyen NT (2001) Using distance functions to solve representation choice problems. Fundam Inform 48:295–314
7. Dang DT, Phan HT, Nguyen NT, Hwang D (2021) Determining 2-Optimality consensus for DNA structure. In: IEA/AIE, vol 2021, pp 427-438
8. Workbench (2016) Manual for CLC genomics workbench. Workbench

9. Holland JH (1975) Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. University of Michigan Press
10. Yang XS (2020) Nature-Inspired Optimization Algorithms. Academic Press
11. Dang D-C et al (2018) Escaping local optima using crossover with emergent diversity. IEEE Trans Evol Comput 22(3):484–497
12. Benito-Parejo M, Merayo MG, Nunez M (2020) An Evolutionary Technique for Supporting the Consensus Process of Group Decision Making. In: IEEE SMC 2020, pp 2201–2206
13. Felling T (2021) Development of a Genetic Algorithm and its Application to a Bi-Level Problem of System Cost Optimal Electricity Price Zone Configurations. Energy Econ 101:105422. https://doi.org/10.1016/j.eneco.2021.105422
14. Husbands P et al (2021) Recent Advances in Evolutionary and Bio-inspired Adaptive Robotics: Exploiting Embodied Dynamics. Appl Intell 51(9):6467–6496
15. Kabir R, Islam R (2019) Chemical reaction optimization for RNA structure prediction. Appl Intell 49(2):352–375
16. Dumitrescu D, Lazzerini B, Jain LC, Dumitrescu A (2000) Evolutionary Computation. CRC Press
17. Maxim AD (2020) Archived elitism in evolutionary computation: Towards improving solution quality and population diversity. Int J Bio-Inspir Com 15(3):135–146
18. Eiben AE, Schippers CA (1998) On evolutionary exploration and exploitation. Fundam Informaticae 35(1-4):35–50
19. Črepinšek M, Liu S-H, Mernik M (2013) Exploration and exploitation in evolutionary algorithms. ACM Comput Surv 45(3):1–33
20. Herrera F, Lozano M (1996) Adaptation of genetic algorithm parameters based on fuzzy logic controllers. Genetic algorithms and soft computing. Physica, Heidelberg, pp 95–125
21. Maleszka M, Nguyen NT (2015) Integration computing and collective intelligence. Expert Syst Appl 42(1):332–340
22. Ji L, Tong S, Li H (2021) Dynamic Group Consensus for Delayed Heterogeneous Multi-agent Systems in Cooperative-competitive Networks via Pinning Control. Neurocomputing 433:1–11
23. Boeck KD, Castellani C, Elborn JS (2014) Medical consensus, guidelines, and position papers: a policy for the ECFS. J Cyst Fibros 13(5):495–498
24. Dang DT, Nguyen NT, Hwang D (2020) A Quick Algorithm to Determine 2-Optimality Consensus for Collectives. IEEE Access 8:221794–221807
25. Ilinkin I, Ye J, Janardan R (2010) Multiple structure alignment and consensus identification for proteins. BMC Bioinform 11(1):71–90
26. Felsenstein J (1997) An alternating least squares approach to inferring phylogenies from pairwise distances. Syst. Biol. 46(1):101–111
27. Amir A, Landau GM, Na JC, Park H, Park K, Sim JS (2011) Efficient Algorithms for Consensus String Problems Minimizing both Distance Sum and Radius. Theor Comput Sci 412(39):5239–5246
28. Hashim FA, Mabrouk MS, Al-Atabany W (2019) Review of different sequence motif finding algorithms. Avicenna J Med Biotechnol 11(2):130–148
29. Lawrence CE, Reilly AA (1990) An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. Proteins Struct Funct Genet 7(1):41–51
30. Tran NTL, Huang C-H (2014) A survey of motif finding web tools for detecting binding site motifs in chIP-seq data. Biol Direct 9(1). https://doi.org/10.1186/1745-6150-9-4
31. Zare-Mirakabad F, Ahrabian H, Sadeghi M, Hashemifar S, Nowzari-Dalini A, Goliaei B (2009) Genetic algorithm for dyad pattern finding in DNA sequences. Genes Genet Syst 84(1):81–93
32. Eskin E, Pevzner PA (2002) Finding composite regulatory patterns in DNA sequences. Bioinformatics 18(1):354–363
33. Bin Ashraf F, Shafi MSR (2020) MFEA: An evolutionary approach for motif finding in DNA sequences. Inform Med Unlocked 21:1–9
34. Koumousis VK, Katsaras CP (2006) A Saw-tooth Genetic Algorithm Combining the Effects of Variable Population Size and Reinitialization to Enhance Performance. IEEE Trans Evol Comput 10(1):19–28
35. Lozano M, Herrera F, Cano JR (2008) Replacement strategies to preserve useful diversity in Steady-State genetic algorithms. Inf Sci 178(23):4421–4433
36. Hutter M, Legg S (2006) Fitness uniform optimization. IEEE Trans Evol Comput 10(5):568–589
37. Shojaedini E, Majd M, Safabakhsh R (2019) Novel adaptive genetic algorithm sample consensus. Appl Soft Comput J 77:635–642
38. Srinivas M, Patnaik LM (1994) Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans Syst Man Cybern 24(4):656–667
39. Amor HB, Rettinger A (2005) Intelligent exploration for genetic algorithms: Using Self-Organizing maps in evolutionary computation. In: GECCO, vol 2005, pp 1531–1538
40. Chow CK, Yuen SY (2011) An evolutionary algorithm that makes decision based on the entire previous search history. IEEE Trans Evol Comput 15(6):741–769
41. Rani S, Suri B, Goyal R (2019) On the effectiveness of using elitist genetic algorithm in mutation testing. Symmetry 11(9):1145. https://doi.org/10.3390/sym11091145
42. Anton VE (2018) On proportions of fit individuals in population of Mutation-Based evolutionary algorithm with tournament selection. Evol Comput 26(2):269–297
43. Chu TH, Nguyen QU, O'Neill M (2018) Semantic tournament selection for genetic programming based on statistical analysis of error vectors. Inf Sci 436–437:352–366
44. Du H, Wang Z, Zhan W, Guo J (2018) Elitism and distance strategy for selection of evolutionary algorithms. IEEE Access 6:44531–44541