Journal of
Computational and Theoretical Nanoscience
Vol. 7, 813–819, 2010

An Efficient Genetic Algorithm Based on the Cultural Algorithm Applied to DNA Codewords Design

Yanfeng Wang*, Ying Niu, Guangzhao Cui, and Xuncai Zhang

College of Electrical and Electronic Engineering, Zhengzhou University of Light Industry, Zhengzhou, 450002, P. R. China

The DNA encoding problem, which has been proved to be an NP hard problem, is one of the key problems for DNA computing, and is usually solved by optimization algorithms. A new efficient genetic algorithm based on the cultural algorithm for the design of DNA codewords is presented in this paper. In this hybrid optimization method, to abstract and manage the information efficiently, the conventional genetic algorithm is combined with the dual evolutionary frame of the cultural algorithm to guide the evolution of the population space with the evolutionary information. Simulation results show this method is convenient for users to design and select proper DNA codewords *in silico*.

Keywords: DNA Computing, DNA Codeword, Genetic Algorithm, Cultural Algorithm.

1. INTRODUCTION

In 1994, Adleman first demonstrated the use of DNA computational solutions to combinatorial problems (Hamilton Path Problem, HPP).[1] The greatest merit of DNA computing is that makes use of the huge memory capacity of DNA molecules and massive parallelism of chemical reactions.[2] Because DNA computing relies heavily on biochemical reactions and is restricted by technological difficulties, it may result in undesirable computations. In this regard, the quality of the DNA code design is playing a critical role in the fidelity of the computation. The problem of designing sets of DNA sequences, which hybridize in a predefined way, is fundamental not only for DNA computing but also for many applications in molecular biology, bioinformatics and DNA nanotechnology such as strand selection, polymerase chain reactions (PCR), DNA-chip arrays, and self-assembly of DNA.

The codewords design problem for DNA computing consists of mapping the instances of an algorithmic problem in a systematic manner onto specific molecules so that the underlying chemical reactions avoid all the sources of error, and the resulting products contain, with a high degree of reliability, enough molecules encoding the answers to the problem's instances to enable a successful extraction.[3] The design of codewords is a bothersome task as the encoding problem is an NP hard problem,[3,4] so we have to settle for less than optimal alternative methods, thus the heuristic approaches maybe the most natural and optimal methods to solve the DNA codewords design problem.

In this paper, we propose a hybrid optimization method based on a genetic algorithm (GA) and the cultural algorithm (CA) for the design of DNA codewords. The paper is organized as follows. Section 2 introduces briefly the problem of DNA codewords design. In Section 3, after discussing the genetic algorithm and cultural algorithm, respectively, the hybrid genetic cultural algorithm (HGCA) is presented. How the hybrid optimization method is applied to DNA codewords design, as well as the simulation experiments, are presented in Section 4. Finally, conclusions are drawn in Section 5.

2. DNA CODEWORDS DESIGN

For error-free and efficient DNA computation, the design of DNA codewords focuses on every DNA molecule can be recognized exclusively.

2.1. The Problem of DNA Codewords Design

The problem of DNA codewords design for computation can be described as the following decision problem.[3]
DNA ENCODING(τ)

*Author to whom correspondence should be addressed.

• *Instance*: A finite set S of n-mers over the nucleic acid bases {A, T, G, C}, a positive integer K, and a mapping $\tau : \Sigma^* \rightarrow Z^+$

• *Question*: Is there a subset $C \subseteq S$ such that $\forall s_i, s_j \in C, \tau(s_i, s_j) \geq k$?

The function τ reflects quality criteria of a given oligonucleotide, (e.g., Hamming distance, the H-distance, or chemical-thermodynamical parameters) to contribute to the solution.

Many studies have attempted to design DNA codewords *in vitro*. Hartemink et al. implemented the exhaustive search method to generate sequences for the programmed mutagenesis.[5] Penchovsky and Ackermann designed DNA sequences by a random search algorithm.[6] Frotus et al. proposed the template method for DNA codewords design.[7] Marathe et al. used a dynamic programming approach to design DNA codewords based on Hamming distance and free energy.[8] Feldkamp demonstrated a DNA sequences compiler algorithms for the design of DNA codewords.[9]

Unlike the above systems, more and more intelligent optimization methods have been offered recently to design DNA codewords. A DNA-based GA was proposed as an application of an evolutionary program searching for good DNA codewords.[10] Wood and Chen proposed and implemented a sequences design scheme suited to the royal road problem using GA.[11] Cui et al. used an improved particle swarm optimization (PSO) algorithms to find good DNA codewords.[12] Wang et al. presented an improved Hopfield neural network algorithm for DNA codewords design.[13] Zhang et al. designed DNA codewords by a tabu search algorithm.[14]

2.2. The Mathematical Model of DNA Codewords Design

The encoding problem can be regarded as a constrained multi-objective optimization problem, and can be solved by using an objective evolutionary method.[15] Before giving the mathematical model of DNA codewords design, the constraints considered in codewords design are discussed as follows.

2.2.1. Encoding Constraints and Analysis

In a word, a good set of DNA codewords should ensure that the following chemical reactions are specific hybridizations, that the controllable PCR can amplify the resulting products, and the resulting products are reliable and can be extracted successfully.[16]

In order to ensure that chemical reactions are controllable, some constraints such as similarity, H-measure,[7] secondary structure,[5, 7] continuity,[7, 17] free energy,[5, 17] melting temperature,[18] GC content,[19] and so on, which have been proposed according to the definition of the encoding problem, are required. All the constraints focus on designing better DNA codewords to reduce the possibility of undesirable chemical reactions.[4] In theory, the H-measure proposed by Garzon can reduce the similarity between two codewords the most. But as the number of codewords increases, Garzon's method requires exponentially increasing time. Otherwise the value of the least distance, which is usually affected by temperature and the reaction conditions is still uncertain. As the criteria always influence and restrict each other, it is irresponsible to use only the distance constraint. In this paper, we mainly consider the following constraint criteria, where the distance constraint is considered as the objective function, and others are regarded as the constraint conditions.

2.3. Distance Constraint

Encoding distance is a parameter to describe the similarity between any two codewords. The greater the encoding distance, the less the similarity. Distance coding has its origin in coding theory, a field of uses the information theory. Its mathematical formulation is Hamming distance. We adopt H-measure which is proposed by Garzon.[20] It is defined as the minimum Hamming distance of any two sequences shifted k $(-n < k < n)$ positions.

$$|x_i, x_j| := \min_{-n<k<n} H(x_i, \sigma^k(\overline{x_j})) \tag{1}$$

where $H(x_i, \sigma^k(\overline{x_j}))$ denotes the Hamming distance, σ^k denotes the right (left) shift in case of $k > 0$ $(k < 0)$, k denotes the number of the shift, and $\overline{x_j}$ denotes the Watson-Crick complementary pair. The corresponding objective function will be given below.

2.4. Continuity Constraint

Secondary structures are usually formed by the interaction of single stranded DNA. Secondary structure includes internal loop, hairpin loop, and bulge loop. To predict the secondary structure, one can simply calculate the Hamming distance of given sequences by folding the sequences to hybridize with themselves. Continuity tests the repeated run of identical bases. If one base is repeated, an unusual secondary structure can be formed. Thus, to avoid forming unexpect secondary structures, for every DNA sequence, the number of times where the repeated bases appear must be in a given range:

$$\sum_{j=1}^{n} (j-1) N_j^{(i)} < d_1 \tag{2}$$

where $N_j^{(i)}$ denotes the number of repeated bases appearing j-times continuously in the sequence x_i and d_1 is defined by users.

2.5. GC Content Constraint

GC content is the percentage of G and C in a sequence, and tightly correlated to the melting temperature and the reaction conditions. To effectively reduce the probability of a non-specific hybridization occurring, GC content must be in a given range:

$$|GC(x_i) - 50\%| < d_2 \tag{3}$$

where $GC(x_i)$ is the GC content of the sequence x_i and d_2 is defined by users.

2.6. Melting Temperature (T_m) Constraint

T_m is defined as the temperature at which 50% of the oligonucleotides and their perfect complements in duplex are denatured. T_m is an important factor in the efficiency of the reaction. The accurate prediction of T_m is particularly critical in the case of the PCR. Large errors in the T_m estimation can lead to the amplification of non-specific products or to an inappropriate hybridization performance in general. For uniform melting temperature, the T_m must be in a given range:

$$|T_m(x_i) - T_m^{'}| \leq d_3 \tag{4}$$

where $T_m^{'}$ is the target melting temperature and d_3 is defined by users.

It has been demonstrated that the method and thermodynamic parameters provided by SantaLucia have a good performance in predicting the experimental T_m of short single-stranded DNA sequences. The T_m calculation is performed according to the following equation.

$$T_m = \frac{\Delta H^\circ}{R \ln(C_T/\alpha)} + \Delta S^\circ \tag{5}$$

where ΔS° and ΔH° denote entropy change and enthalpy change under a certain temperature between every base, respectively.[21]

2.6.1. Objective Function

The objective function consists of two modules: the maximizing non-cross-hybridization module and the minimizing cross-hybridization module. The maximizing non-cross-hybridization module makes the similarity of a set of codewords as small as possible, and the latter reduces the probability of cross-hybridization occurring.

2.7. Maximizing Non-Cross-Hybridization

To maximize the probability of occurring non-cross-hybridization between any two codewords, we propose the following evaluation term.

$$f_{\text{sim}}(X) = \max_{i,j,i<j} \max_{-n<k<n} \{n - H(x_i, \sigma^k(x_j))\} \tag{6}$$

$$f_{\text{sim}}(x_i, x_j) = \max_{-n<k<n} \{n - H(x_i, \sigma^k(x_j))\} \tag{7}$$

where $f_{\text{sim}}(X)$ is the similarity measure of DNA codewords set X and $f_{\text{sim}}(x_i, x_j)$ is the similar similarity measure of the codewords x_i, x_j. $H(x_i, \sigma^k(x_j))$ denotes the number of different bases in sequences x_i and x_j.

2.8. Minimizing Cross-Hybridization

To minimize the probability of occurring cross-hybridization between any two codewords, we propose the following evaluation term.

$$f_{\text{cross}}(X) = \max_{i,j,i<j} \max_{-n<k<n} \{n - H(x_i, \sigma^k(\overline{x_j}))\} \tag{8}$$

$$f_{\text{cross}}(x_i, x_j) = \max_{-n<k<n} \{n - H(x_i, \sigma^k(\overline{x_j}))\} \tag{9}$$

where $f_{\text{cross}}(X)$ is the cross-hybridization probability measure of set X and $f_{\text{cross}}s(x_i, x_j)$ is a constraint of the cross-hybridization probability between sequences x_i and x_j.

We formulate the objective function as a minimization problem, and use the weighted sum to deal with the selected constraints.

3. GENETIC CULTURAL ALGORITHM

3.1. Genetic Algorithm

The concept of a GA was developed by Holland in 1975,[22] GAs are powerful tools in solving search and optimization problems. It originates from the idea of natural selection and natural genetic process, combines the survival of the fittest and stochastic information change mechanism of chromosomes in the group. Because of its robustness, GAs have been successfully adopted in many complex optimization problems and show their merits over traditional optimization methods, especially when the system under study has multiple local optimum solutions.[23]

In biology, a niche is a role an organism plays in its environment. It encompasses all relationships that the organism (or population) has with its environment and with other organisms and populations in its environment. In order to keep the population's variety, avoid trapping into local extremum, here we apply the biological concept of the niche to GA.

We adopt the niche method based on a sharing function. The main idea is to regulate the fitness in order to keep exceptional individuals of the population increasing significatively, because every individual's genetic probability is controlled by the fitness. So keep the population's variety and create a niche evolutionary environment.

DEFINITION 1. The function which denotes the affinity between two individuals in the population is called the sharing function, written as $S(d_{ij})$, where d_{ij} means some relationship between i and j.

DEFINITION 2. The summation of the sharing function's value between one individual and other individuals in the population is called the sharing degree, written as S_i, is the degree of one individual sharing in the population, that is, $S_i = \sum_{j=1}^{M} S(d_{ij})$ $(i = 1, 2, \ldots, M)$. After computing every individual's sharing degree, every individual's fitness is regulated according to the following formula.

$$F_i^{'}(X) = F_i(X)/S_i, \quad (i = 1, 2, \ldots, M) \tag{10}$$

3.2. Cultural Algorithm

In human society, a culture can be viewed as a vehicle for the storage of information that is potentially accessible to all members of the society, and that can be useful in guiding problem-solving activities. Originated by this idea, Reynolds developed a CA in 1994.[24]

The CAs operate on two spaces: a population space and a belief space. First, they operate on the population space where a set of individuals (called population) is adopted. Each individual has a set of features independent from each other which allows us to determine its fitness. Through time, such individuals can be replaced by some of its descendants, obtained after applying a set of operators to the population. The second space is the belief space, in which the knowledge acquired by the individuals along the evolutionary process is stored. The information contained in this space must be accessible to any individual, so that it can use it to modify its behavior.[25]

To unify both spaces, a communication protocol is established such that it dictates rules regarding the type of information to be exchanged between these two spaces. For example, to update the belief space, the individual experiences of a select set of individuals are incorporated. This select group of individuals is obtained with the function acceptance which is applied to the entire population. On the other hand, the operators that modify the population (i.e., recombination and mutation) and the selection operator are modified by the function influence. This function acts in such a way that the individuals resulting from the application of the operators tend to approach the desirable behavior while staying away from undesirable behaviors. Such desirable and undesirable behaviors are defined in terms of the information stored in the belief space. These two functions are used to establish the

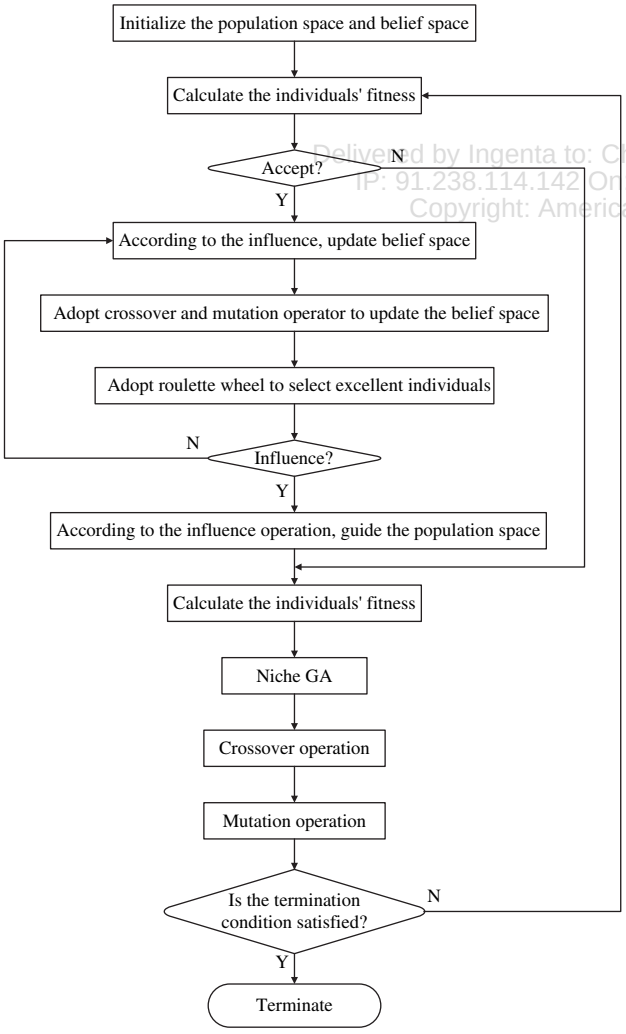

Fig. 1. The flow chart of the HGCA.

Table I. Comparison Results of the Codewords from 27 and Generated by HGCA.

Sequence	H-measure	Continuity	GC Content	T_m	Free Energy
HGCA					
ACAACCGCCC AATATAGGAG	118	9	0.5	63.9811	−26.33
GTGGTCGTAC ATACAAACC	109	9	0.5	63.1849	−26.35
GAATCGATCA CGTAGCTCTG	130	0	0.5	62.4727	−26.04
ACCCGACTTA GGAATGTTCG	125	0	0.5	64.1369	−26.55
GGGTCTCGATA TCTGTCTTG	111	9	0.5	61.5179	−25.31
CTAATTTCGT ACCGTGGGTG	114	18	0.5	62.9276	−26.14
TGTGAGTTAG CCCGGAAACA	111	18	0.5	66.1881	−27.34
Deaton's Codewords [27]					
CTTGTGACCG CTTCTGGGGA	130	16	0.6	69.0571	−28.74
CATTGGCGGC GCGTAGGCTT	108	0	0.65	73.2553	−31.35
ATAGAGTGGA TAGTTCTGGG	122	9	0.45	59.8408	−24.24
GATGGTGCTT AGAGAAGTGG	112	0	0.5	62.3221	−25.63
TGTATCTCGT TTTAACATCC	130	16	0.35	57.301	−23.33
GAAAAAGGAC CAAAAGAGAG	105	16	0.4	58.6868	−23.89
TTGTAAGCCT ACTGCGTGAC	117	0	0.5	64.67	−26.9

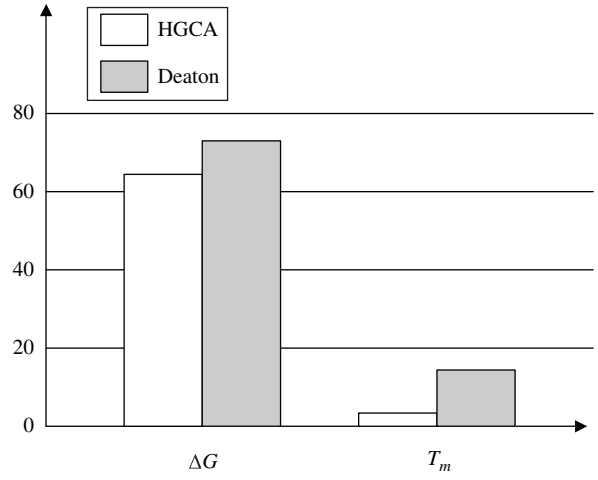

Fig. 2. Comparison results of the codewords from 27 and generated by the HGCA.

communication between the two spaces (i.e., population and belief).[25] For more information about the interactions between these two spaces, please see Ref. [26]

3.3. Hybrid Genetic Cultural Algorithm

For designing DNA codewords, here we use the dual structure of the CA, embed the GA into the cultural frame as an evolutionary process of the population space, establish the main population space and belief space based on GA, and develop the HGCA. Figure 1 shows the flow chart of the HGCA and the details are presented as below:

3.3.1. Evolution of the Population Space

(1) Initialize the population space. Generate the initial population in the whole search space randomly.
(2) Calculate and arrange the individuals' fitness.
(3) Select the individuals whose fitness is great, and transfer them to the belief space.
(4) According to the influence of the belief space, calculate all the individuals' fitness again.
(5) Genetic operation: according to the niche genetic algorithm, adopt the crossover operator and mutation operator.
(6) If the termination condition is not satisfied, return to 2).

3.3.2. Evolution of the Belief Space

(1) Initialize the belief space.
(2) According to the accept function, accept the excellent individuals from the population space.
(3) Adopt the crossover operator and mutation operator to update the belief space, so the belief space can keep

Table II. Comparison Results of the Codewords from 15 and Generated by HGCA.

Sequence	H-measure	Continuity	GC Content	T_m	Free Energy
HGCA					
AATGGGTAGAAGATGGCAGC	194	9	0.5	64.5766	−26.57
CACGGATTTGCGTCTGAACT	195	9	0.5	65.5602	−27.43
TGCAAGTTAAGCTCGTCTCC	199	0	0.5	64.6817	−26.87
CAATTGACCAGTCAGTGCTG	221	0	0.5	63.7282	−26.42
CGGAACAAGAGAAACCTTGC	193	9	0.5	63.9961	−26.61
CTAGCAGGGTGTAGAGCATA	215	9	0.5	62.7281	−25.71
TACTAGTGCCCACACGTCTA	195	9	0.5	64.4213	−26.52
GGCGGAGAATCGTGCATTAA	195	0	0.5	65.4457	−27.36
GTTGGCTCCATTTCTGGAAC	194	9	0.5	63.7675	−26.33
TGGCACGCCTATTTCACATG	206	9	0.5	65.6999	−27.38
CGATGTCAGAGCGATGTTGT	207	0	0.5	65.0508	−27.22
ACATCTCACCGTAACCCAGT	202	9	0.5	65.3444	−26.86
TATGTTCGCTGGATGTACCC	206	9	0.5	63.9461	−26.4
GCACTCCAAACAGCTGCATA	168	9	0.5	65.6407	−27.29
Shin's Codewords Shin2005					
GTGACTTGAGGTAGGTAGGA	213	0	0.5	62.2239	−25.36
ATCATACTCCGGAGACTACC	197	0	0.5	62.1413	−25.42
CACGTCCTACTACCTTCAAC	221	0	0.5	61.9397	−25.55
ACACGCGTGCATATAGGCAA	204	0	0.5	67.2662	−28.22
AAGTCTGCACGGATTCCTGA	205	0	0.5	65.9669	−27.24
AGGCCGAAGTTGACGTAAGA	219	0	0.5	65.9024	−27.35
CGACACTTGAAGCACACCTT	213	0	0.5	65.4594	−27.25
TGGCGCTCTACCGTTGAATT	190	0	0.5	66.8953	−27.9
CTAGAAGGATAGGCGATACG	197	0	0.5	61.0777	−25.22
CTTGGTGCGTTCTGTGTACA	191	0	0.5	65.1612	−27.14
TGCCAACGGTCTCAACATGA	209	0	0.5	66.7991	−27.72
TTATCTCCATAGCTCCAGGC	192	0	0.5	63.1165	−25.84
TGAACGAGCATCACCAACTC	202	0	0.5	64.9647	−27.01
CTAGATTAGCGGCCATAACC	188	0	0.5	62.2436	−25.7

excellent individuals.
(4) Adopt a roulette wheel to select excellent individuals. Judge whether the influence function's condition is satisfied. If it is satisfied, then according to the influence operation guide the population space, otherwise return to 3, update the belief space again.

4. SIMULATION EXPERIMENT

In this section, the HGCA algorithm is implemented to show the performance of our system by comparing our algorithm with other DNA codewords design systems. With respect the model of codewords design proposed above, the HGCA algorithm was implemented on a PC using the C++ programming language. The population size, maximum generation number and the length of DNA codewords were selected 20, 500 and 20, respectively. The probability of crossover and mutation rate were set to 0.6 and 0.05, respectively.

We compared the codewords generated randomly by HGCA algorithm with.[15, 27] In Ref. [27], Deaton et al. used 7 codewords with length 20 (as shown in Table I) generated by a GA to solve the HPP problem. We generated randomly the same number of codewords with the same length using the HGCA algorithm, and evaluated these two sets of DNA codewords using the evaluation terms proposed in Ref. [16] The comparison results are shown in Table I and Figure 2.

From Table I and Figure 2, we can see that our codewords are much better than the codewords from,[27] because the T_m of our sequences is comparatively identical. Our sequences show much lower free energy. This means the sequences generated by the HGCA have much more advantages in keeping a uniform melting temperature and lower probability of non-specific hybridizations occurring.

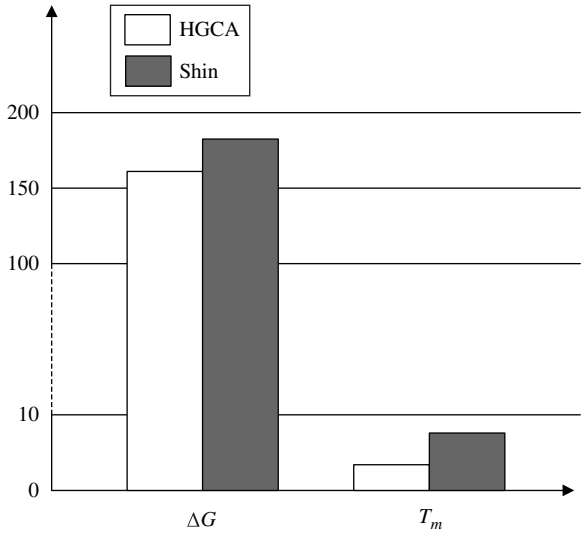

Fig. 3. Comparison results of the codewords from 15 and generated by the HGCA.

It further indicates a much higher probability of hybridization with the correct complementary sequences.

Shin et al. formulated the DNA codewords design as a multi-objective optimization problem and solved it using a constrained multi-objective evolutionary algorithm in Ref. [15]. Here we abstracted 14 codewords with length 20 from,[15] and compared them with the codewords generated by the HGCA algorithm. The comparison results are shown in Table II and Figure 3. We can see that in regard to T_m and free energy, our sequences are much better than the sequences from.[15]

As a result, we can see that the codewords generated by the HGCA algorithm are effective. The HGCA algorithm is better than the algorithms[15, 27] in keeping a uniform melting temperature and preventing non-specific hybridizations.

5. CONCLUSIONS

In this paper, we have proposed a hybrid optimization method based on GAs and the CA for designing suitable DNA codewords satisfying the definition of the encoding problem in DNA computing. The HGCA embeds a GA into the cultural frame and sets up the main population space and belief space based on the GA. This manages the information efficiently, guides the evolution of the population space with the evolutionary information, and improves the search efficiency. The HGCA provides a way of solving the DNA encoding problem. The DNA codewords designed by the HGCA have been compared with those designed by other existing sequence design systems. The results show the feasibility and validity of this method.

Acknowledgment: The authors appreciate the editors and the anonymous referees for their helpful comments, as well as the financial support from National Scientific Foundation of China under grants 60773122, 60573190, and the Innovation Scientists and Technicians Troop Construction Projects of Henan Province.

References

1. L. M. Adleman, *Science* 266, 1021 **(1994)**.
2. R. S. Braich, N. Chelyapov, C. Johnson, and P. W. K. Rothemund, *Science* 296, 499 **(2002)**.
3. M. Garzon, R. Deaton, P. Neathery, D. R. Franceschetti, and S. E. Steve, On the encoding problem for DNA computing, *Preliminary Proceedings 3rd DIMACS Workshop on DNA Based Computers,* University of Pennsylvania, USA, **(1997)**, pp. 230–237.
4. J. Sager and D. Stefanovic, Designing nucleotide sequences for computation: a survey of constraints, *Preliminary Proceedings of the 11th International Meeting on DNA Computing*, London, ON, Canada **(2005)**, pp. 275–289.
5. A. J. Hartemink, D. K. Gifford, and J. Khodor, *Biosystems* 52, 227 **(1999)**.
6. R. Penchovsky and J. Ackermann, *J. Comput. Bio.* 10, 215 **(2003)**.
7. A. G. Frutos, Q. Liu, A. J. Thiel, A. M. W. Sanner, A. E. Condon, L. M. Smith, and R. M. Corn, *Nucleic Acids Res.* 25, 4748 **(1997)**.

8. A. Marathe, A. E. Condon, and R. M. Corn, *J. Comput. Bio.* 8, 201 **(2001)**.

9. U. Feldkamp, W. Banzhaf, and H. Rauhe, A DNA sequence compiler, *Proceedings of the 6th International Workshop DNA-Based Computers*, Leiden, Netherlands **(2000)**, p. 253.

10. R. Deaton, R. C. Murphy, J. A. Rose, M. Garzon, D. R. Franceschetti, and S. E. Stevens, Jr., A DNA based implementation of an evolutionary search for good encodings for DNA computation, *Proceedings of IEEE International Conference on Evolutionary Computation*, Indianapolis, IN, USA **(1997)**, pp. 267–271.

11. D. H. Wood and J. Chen, Physical separation of DNA according to royal road fitness, *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington, DC, USA **(1999)**, pp. 1016–1025.

12. G. Cui, Y. Niu, Y. Wang, X. Zhang, and L. Pan, *Progress in Natural Science* 17, 712 **(2007)**.

13. Y. Wang, G. Cui, X. Zhang, and B. Huang, *J. Comput. Theor. Nanosci.* 4, 1257 **(2007)**.

14. K. Zhang, J. Xu, X. Geng, J. Xiao, and L. Pan, *Progress in Natural Science* 18, 623 **(2008)**.

15. S.-Y. Shin, I.-H. Lee, D. Kim, and B.-T. Zhang, *IEEE Transaction on Evolutionary Computation* 9, 143 **(2005)**.

16. Y. Wang, G. Cui, and B. Huang, *Dynamics of Continuous, Discrete and Impulsive Systems, Series B* S1, 3410 **(2006)**.

17. F. Tanaka, M. Nakatsugawa, M. Yamamoto, T. Shiba, and A. Ohuchi, Developing support system for sequence design in DNA computing, *Proceedings of the 7th International Workshop DNA-Based Computers*, Tampa, FL, USA **(2001)**, pp. 340–349.

18. M. Arita and S. Kobayashi, *New Generation Computer* 20, 263 **(2002)**.

19. U. Feldkamp, S. Saghafi, and H. Rauhe, DNA sequence generator: A program for the construction of DNA sequences, *Proceedings of the 7th International Workshop DNA-Based Computers*, Tampa, FL, USA **(2001)**, pp. 179–188.

20. M. Garzon, R. Deaton, P. Neathery, R. C. Murphy, S. E. Stevens, Jr., and D. R. Franceschetti, A new metric for DNA computing, *Proceedings of the 2nd Annual Genetic Programming Conference*, Stanford University. San Francisco, CA **(1997)**, pp. 472–487.

21. F. Tanaka, A. Kameda, M. Yamamoto, and A. Ohuchi, *Biochemistry* 43, 7143 (2004).

22. J. H. Holland, *Adaptation in Natural and Artificial Systems*. Univ. of Michigan Press, Ann Arbor **(1975)**.

23. K. Deepa and M. Thakur, *Applied Mathematics and Computation* 188, 895 **(2007)**.

24. R. G. Reynolds, An introduction to cultural algorithms, *Proceedings of the 3rd annual Conference on Evolution Programming*, San Diego, California **(1994)**, pp. 131–139.

25. C. A. C. Coello and R. L. Becerra, Evolutionary multiobjective optimization using a cultural algorithm, *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, Indianapolis, USA **(2003)**, pp. 6–13.

26. R. G. Reynolds, New Ideas in Optimization, edited by D. Corne, M. Dorigo, and F. Glover, McGraw-Hill, London **(1999)**, pp. 367–377.

27. R. Deaton, R. C. Murphy, M. Garzon, D. R. Franceschetti, and J. S. E. Stevens, Good encodings for DNA-based solutions to combinatorial problems, *Proceedings of the 2nd DIMACS Workshop on DNA-based Computers*, Princeton University, USA **(1996)**, pp. 159–171.

Received: 12 November 2008. Accepted: 3 February 2009.