A new approach based on PSO algorithm to find good computational encoding sequences*

Cui Guangzhao[1], Niu Yunyun[1]**, Wang Yanfeng[1], Zhang Xuncai[2] and Pan Linqiang[2]

(1. School of Electrical and Electronic Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China; 2. Research Institute of Biomolecular Computer, Huazhong University of Science and Technology, Wuhan 430074, China)

Accepted on November 30, 2006

Abstract Computational encoding DNA sequence design is one of the most important steps in molecular computation. A lot of research work has been done to design reliable sequence library. A revised method based on the support system developed by Tanaka et al. is proposed here with different criteria to construct fitness function. Then we adapt particle swarm optimization (PSO) algorithm to our encoding problem. By using the new algorithm, a set of sequences with good quality is generated. The result also shows that our PSO-based approach could rapidly converge at the minimum level for an output of the simulation model. The celerity of the algorithm fits our requirements.

Keywords: DNA computing, computational encoding DNA sequences, PSO algorithm, fitness function.

DNA computing relies on biochemical reactions of DNA molecules and may result in incorrect computations if the DNA sequences have not good quality. That is why codeword design is considered as one of its most important steps. DNA sequences used in DNA computing must satisfy several constraints, which focus on the design of DNA sequences that reduce the possibility of undesirable reactions.

In the published literatures[1—4], the constraints H-measure, similarity, continuity, melting temperature, GC content and free energy have often been mentioned to select good DNA sequences. Tanaka et al. have developed a support system for sequence design in DNA computing. They offered some sequence fitness criteria, and generated sequences using simulated annealing[5]. Here, we use the similar method with different criteria to construct more reasonable evaluation function. Then we consider it as fitness function in our adapted particle swarm optimization (PSO) approach for DNA sequence design.

PSO is a generic heuristic optimization algorithm based on the concept of swarm intelligence. It requires less computation times and less memory. Unlike GA, it has no evolution operators such as crossover and mutation. It is easy to implement and has fewer parameters to adjust. Till now PSO has been used to solve many engineering and economic problems. However, it has not been employed so far in DNA computing. Here, we apply the adapted PSO approach to search for good DNA sequences with fitness function mentioned above.

This paper is organized as follows: fitness function is constructed in Section 1, followed by a brief overview of PSO algorithm and its implementation to DNA encoding. The results and analyses are in Section 4. Section 5 is the conclusion.

1 Constraints formulation used in the fitness function

Many experiments show that randomly generated codes are inadequate for accurate DNA computation. We desire a set of DNA sequences to form stable duplex with their complements. We also need to ensure that two sequences which are not complemented each other do not interact. As in many published literatures, several criteria have been used to estimate the quality of the library. The DNA sequence design problem can be written as[6]:

$$\min F(x) = (f_1(x), f_2(x), \cdots, f_n(x)) \quad (1)$$

where $f_i(x)$ denotes the fitness measure of the constraint such as H-measure, secondary structure, continuity, melting temperature, GC content and so on.

* Supported by National Natural Science Foundation of China (Grant Nos. 60573190, 30370356) and Henan Province Scientific Foundation (Grant Nos. 511011600, 211050900, 2004922025)

** To whom correspondence should be addressed. E-mail: niuyunyun1003@163.com

We tried to accumulate the objectives as many as possible, but it is neither proper nor necessary to consider all of them. For their special relations, we need to choose them carefully. Tanaka et al. gave a method to calculate correlation coefficients of them, and then predigested the constraint set. Based on that approach, we choose four constraints, distance, continuity, GC content, and T_m to form fitness function. More information about this approach can be obtained in Ref. [5].

1.1 Formulation of distance constraint

The distance constraint of two sequences both complementary and parallel is considered in this paper. It computes how many nucleotides are different in the same direction of two given sequences to keep each sequence as unique as possible, including position shift. And it also calculates how many nucleotides are complementary between the given sequences to prevent cross-hybridization of two sequences. It is defined as follows:

$$F_{HD}(\Sigma) = \sum_{i=1}^{n} f_{HD}(x_i) \tag{2}$$

$$f_{HD}(x_i) = \sum_{j=1}^{n} (HD(x_i, x_j) + HD(x_i, \overrightarrow{x_j})) \tag{3}$$

1.2 Formulation of continuity

Same bases occur continuously (such as "AAAAA") in a sequence may cause unexpected structures. Continuity is an important factor to measure the quality of a sequence. Its formulation is described as[6]:

$$F_{con}(\Sigma) = \sum_{i=1}^{n} f_{con}(x_i) \tag{4}$$

$$f_{con}(x) = \sum_{i=1}^{L-t+1} \sum_{\alpha \in \Lambda} T(c_\alpha(x, i), t)^2 \tag{5}$$

$$C_\alpha(x, i) = \begin{cases} c, & \text{if } \exists c, \text{ s.t. } x^i \neq \alpha,\ x^{i+j} = \alpha \\ & \text{for } 1 \leqslant j \leqslant c,\ x^{i+j+1} \neq \alpha; \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

1.3 Formulation of GC content

The percentage of G and C in a sequence can affect chemical properties of DNA sequences. Well-defined GC content can keep the melting temperature uniform, and reduce the probability of occurring non-specific hybridization effectively. The formulation is

$$F_{GC}(\Sigma) = \sum_{i=1}^{n} f_{GC}(x_i) \tag{7}$$

$$f_{GC}(x_i) = (GC(x_i) - GC_{\text{defined}})^2 \tag{8}$$

1.4 Formulation of T_m

The melting temperature is the temperature in equilibrium at which 50% of the oligonucleotides have hybridized to their perfect complements and 50% of the oligonucleotides are separated. All of the sequences in the library need to have similar melting temperatures or melting temperatures above some threshold. There are many equations to calculate melting temperature such as the Wallace 2—4 rule[7], the GC% method[8], and the nearest-neighbor model[9]. Here, we will choose the nearest-neighbor model. The description of this measure is

$$F_{T_m}(\Sigma) = \sum_{i=1}^{n} f_{T_m}(x_i) \tag{9}$$

$$f_{T_m}(x_i) = (T_m(x_i) - T_m(\Sigma))^2 \tag{10}$$

$$T_m = \frac{\Delta H^0}{R \ln(C_T/\alpha)} + \Delta S^0 \tag{11}$$

where $T_m(x_i)$ is the melting temperature of the generated sequence, $T_m(\Sigma)$ is the target melting temperature, R is the gas constant, C_T is the concentration, ΔH^0 is the enthalpy and ΔS^0 is the entropy. Parameter α is set to be 1 for self-complementary and be 4 for non-self-complementary. Parameters ΔH^0 and ΔS^0 are calculated based on the nearest-neighbor model[9, 10].

1.5 Construct fitness function

We normalize all the evaluation terms above, and then get the contribution ratio as shown in Table 1. More detail information about this process can be obtained in Ref. [5].

Table 1. Contribution ratio

f_i	f_{HD}	f_{GC}	f_{con}	f_{T_m}
w_i	0.3242	0.1347	0.3100	0.2311

The fitness function can be described as:

$$f = \sum_i w_i f_i,$$
$$i \in \{\text{distance, } GC \text{ content, } T_m, \text{ continuity}\} \tag{12}$$

2 PSO algorithm and its implementation to DNA encoding

2.1 A brief overview of PSO algorithm

PSO algorithm was introduced by Kennedy and

Eberhart in 1995[11,12]. It is motivated from simulation of the behavior of social systems such as fish schooling and birds flocking. In PSO, the system is initialized with a population of random solutions and searches for optima by updating generations. The potential solutions, called particles, fly through the problem space by following the current optimum particles.

PSO algorithm for N-dimensional problem formulation can be described as follows: Let X be the particle position and V be its speed in a searched space. Consider i as the particle in the total swarm. Now the position of the ith particle can be represented as $X_i = (x_{i1}, x_{i2}, \cdots, x_{iN})$ in the N-dimensional space. All the positions are evaluated by a fitness function. The best previous position of the ith particle is stored and represented as *pbcst*. The best value among all *pbest* is represented as *gbest*. The ith particle velocity is represented as $V_i = (v_{i1}, v_{i2}, \cdots, v_{iN})$. The kth iteration of the individual position and its velocity can be formulated as equations below:

$$v_{ij}^{k+1} = \omega * v_{ij}^{k} + c_1 rand() * (pbest_{ij}^{k} - x_{ij}^{k}) + c_2 rand() * (gbest_{j}^{k} - x_{ij}^{k}) \quad (13)$$

$$x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1},$$

$$i = 1, 2, \cdots, S;\ j = 1, 2, \cdots, N \quad (14)$$

where N is the number of dimensions in a particle, S is the number of particles, ω is the inertia weight factor, c_1 and c_2 are the acceleration constants, $rand_1()$ and $rand_2()$ are the uniform random values in the range [0, 1], v_{ij}^{k} is the velocity of the jth dimension in the ith particle, x_{ij}^{k} is the current position of the jth dimension in the ith particle at iteration k. Fig.1 describes a particle's movement in the 2-dimensional space.

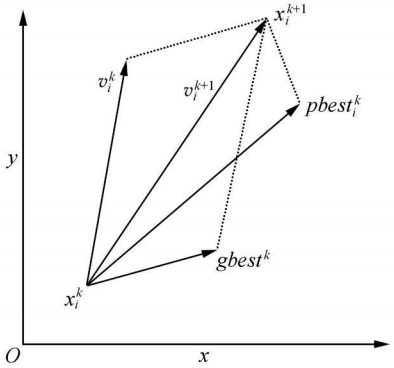

Fig. 1. A particle's movement in the two-dimensional space.

In this paper, we decrease inertia weight ω linearly from the maximum ω_{max} to the minimum ω_{min} during a run[13]. Its value is set according to the following equation:

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{iter_{max}} * iter \quad (15)$$

where $iter_{max}$ denotes the maximum iteration number, and *iter* denotes the current iteration number. The pseudo code of the basic PSO algorithm is as follows:

```
For each particle {
   Initialize particle;
}
While maximum iterations or minimum criteria is not attained{
   For each particle {
      Calculate fitness value;
      If (fitness value < pbest){
      Update pbest;
      If (pbest < gbest) Update gbest;
      }
   }
   For each particle{
      Calculate particle velocity v according to Eq. (13);
      If (v > Vmax) v=Vmax;
      Else if (v<-Vmax) v=-Vmax;
      Calculate particle position x according to Eq. (14);
      If (x > Xmax) x=Xmax;
      Else if (x<-Xmax) x=-Xmax;
   }
}
```

2.2 Implementation of PSO method in DNA encoding

This paper presents a quick solution to search for good DNA sequences using the PSO algorithm. Twenty 20-mer DNA sequences are connected one by one in the same direction to form a 400-mer DNA strand. We denote it as a particle, and then ten strands like that form a particle swarm. We define A=0, C=1, G=2, T=3, and give an example of a 20-mer DNA sequence in Fig.2.

Fig. 2. An example of a 20-mer DNA sequence (here A=0, C=1, G=2, T=3).

Inertia weight linearly decreases from 0.9 to

0.4 during a run using Eq. (11). For this discrete problem, we let $X_i=[X_i] \bmod 4$ in step 5. The detail steps are given bellow.

Step 1: Initialize particle positions and velocities. Randomly generate particles X_1, X_2, …, X_{10}. Each of them is a 400-mer DNA sequence. The particle velocities V_1, V_2, …, V_{10} are also generated randomly, where $v_{ij} \in R$.

Step 2: Use Eq. (12) to calculate all the particles' fitness values.

Step 3: Update *pbest* and then *gbest* based on the values. If the new value is better than the previous *pbest*, the new value is set to be *pbest*.

Step 4: If the stopping criteria are met, the positions of particles represented by *gbest* are the optimal solution. Otherwise, new velocities for all the dimensions in each particle are calculated using Eq. (13).

Step 5: The position of each particle is updated using Eq. (14). Let $X_i=[X_i] \bmod 4$, return to step 2.

The flow chart is described in Fig. 3.

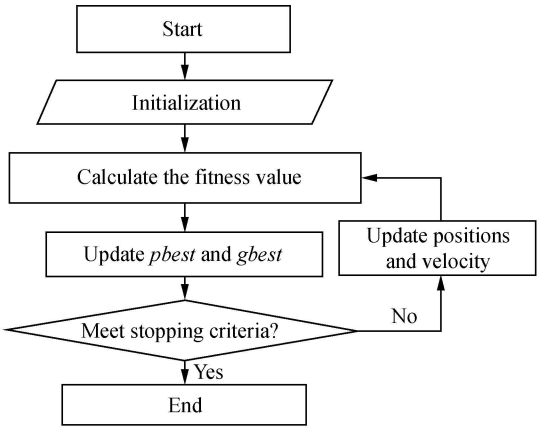

Fig. 3. The flow of PSO algorithm for DNA sequence design.

3 Results and analysis

Finally, we generate a set of good DNA sequences by using adapted PSO algorithm, and evaluate it with several typical criteria to assess the effectiveness of the proposed algorithm. The generated sequences and their evaluation values are listed in Table 2.

Table 2. The generated sequences and their evaluation values

Words	distance	Con	GC	T_m	ΔG	Hairpin
GTCAAATTCCCTCTATCGTC	289	18	0.45	59.9257	−24.58	0
AGCGATAGTAGATCACCTGC	295	0	0.50	63.4162	−26.21	3
CACGATATAGCTTCGTGCCG	270	0	0.55	65.5196	−27.69	22
AATACACCGCTCACCAAGGA	268	0	0.50	65.9901	−27.22	0
AACAGGGAAGAATGCAGAGG	260	9	0.50	64.4354	−26.45	0
CCTCTACCAGCCAATGATGC	284	9	0.55	65.2329	−27.00	0
TTAGGACTCGACGCCACTCC	261	0	0.60	68.1555	−28.50	0
CCATGACCGAGGATCCACGT	256	0	0.60	68.5855	−28.61	0
CGCCATTATCAGGCCTTTAC	290	9	0.50	63.3848	−26.30	3
ACACAGTGGACGCACATACA	305	0	0.50	66.4720	−27.60	3
TTATCCCGCCTCTTCTCCGT	258	9	0.55	67.4031	−27.90	0
AATACGGTTCAAGCGGCTTC	275	0	0.50	65.6572	−27.50	3
TAAAGGCGCGTGATCGGAAG	276	0	0.55	67.5152	−28.50	0
TTGTTCGGGATTGAGCAACT	277	9	0.45	64.6279	−26.70	6
GTCACTGAGTCAGCACTCAT	305	0	0.50	64.0660	−26.50	25
CCATAAACTGCCAGCTCGCG	276	9	0.60	68.7203	−29.10	0
CAACATAGAGTCAGGCGCTG	289	0	0.55	65.4053	−27.40	0
CCAATGAGTCACCTCGTTCG	309	0	0.55	65.3200	−27.30	3
GGGGTGGAGGCCCAACTATT	288	25	0.60	68.7331	−28.20	9
CAGCGGTCTGAACCTCCATA	282	0	0.55	66.0663	−27.40	6

In it for each strand, distance, continuity, GC content, melting temperature, free energy and hairpin are reported. Compared to other sequences from previous publications[14−16], our approach can generate better or comparative strings in all objectives.

Otherwise, the executive time also fits our requirements, and it seems very effective to find the op-

timum solution. At the end of iterations, the fitness value will approximate to be stable. The convergence curve of the algorithm is shown in Fig.4.

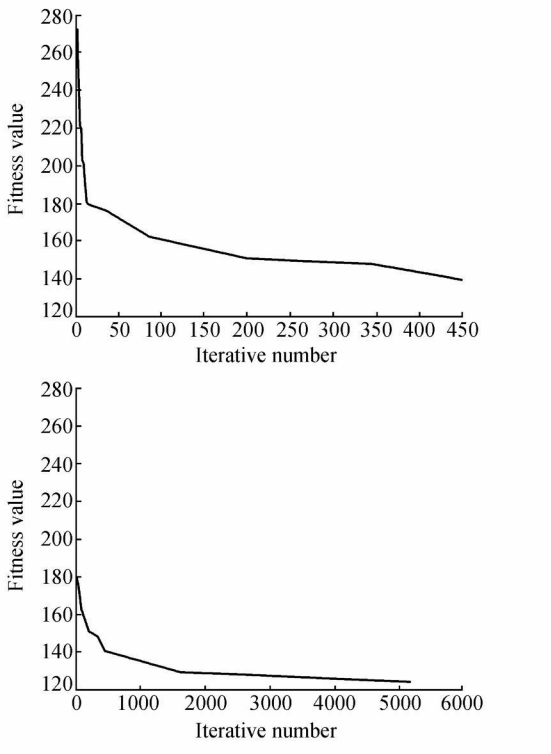

Fig. 4. Convergence curve.

We have been trying to apply our approach to a large scale. But the speed slows down when the sequence library comes big. Now several parameters are being adjusted for better results.

4 Conclusions

In this paper, we try to apply PSO algorithm to produce good sequences for DNA computing. The results show that our adapted PSO algorithm is efficient to generate a set of serials with good quality. And we are also content with its celerity. Although it seems a little rough and still needs some improvements, it has already indicated the advantage of PSO algorithm such as easy implementation and few parameters. The approach will be applied to a larger scale after the parameters are chosen properly.

For the factors controlling DNA computing are still not very clearly known, algorithm improvement is not enough for the entire problem. Much work needs to be done in the future, such as further researching DNA chemistry and exploring accurate model used in DNA sequence design.

References

1 Dirks RM, Lin M, Winfree E, et al. Paradigms for computational nucleic acid design. Nucleic Acids Research, 2004, 32(4): 1392—1402
2 Garzon M and Deaton RJ. Codeword design and information encoding in DNA ensembles. Natural Computing, 2004, 3: 253—292
3 SantaLucia J Jr and Hicks D. The thermodynamics of DNA structural motifs. Annual Review of Biophysics Biomolecular Structure, 2004, 33: 415—440
4 Sager J and Stefanovic D. Designing nucleotide sequences for computation: a survey of constraints. In: Preliminary Proceedings of the 11th International Meeting on DNA Computing. London, Ontario, Canada: Springer LNCS (3892), 2005, 275—289
5 Tanaka F, Nakatsugawa M, Yamamoto M, et al. Developing support system for sequence design in DNA computing. In: Preliminary Proceedings of Seventh International Meeting on DNA Based Computers, Tampa, Florida, USA, 2001, 340—349
6 Shin SY, Lee IH, Kim D, et al. Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing. IEEE Transactions on Evolutionary Computation, 2005, 9: 143—158
7 Wallace RB, Shaffer F, Murphy RF, et al. Hybridization of synthetic oligodeoxyribonucleotides to phichi 174 DNA: The effect of single base pair mismatch. Nucleic Acids Research, 1979, 6(11): 3543—3557
8 Wetmur JG. DNA probes: applications of the principles of nucleic acid hybridization. Critical Reviews in Biochemistry and Molecular Biology, 1991, 26(3/4): 227—259
9 SantaLucia J Jr. A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. In: Proceedings of the National Academy of Sciences of the United States of America, 1998, 95(4): 1460—1465
10 Tanaka F, Kameda A, Yamamoto M, et al. Thermodynamic parameters based on a nearest-neighbor model for DNA sequences with a single-bulge loop. Biochemistry, 2004, 43: 7143—7150
11 Kennedy J and Eberhart R. Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks. Piscataway, NJ, 1995, 5: 1942—1948
12 Eberhart R and Kennedy J. A new optimizer using Particle swarm theory. In: Proceedings of the 6th International Symposium on Micro Machine and Human Science, 1995, 39—43
13 Shi YH and Eberhart RC. Empirical study of particle swarm optimization. In: Proceedings of Congress on Evolutionary Computation. Piscataway, NJ, 1999, 1945—1950
14 Tanaka F, Kameda A, Yamamoto M, et al. Design of nucleic acid sequences for DNA computing based on a thermodynamic approach. Nucleic Acids Research, 2005, 33(3): 903—911
15 Tanaka F, Nakatsugawa M, Yamamoto M, et al. Toward a general-purpose sequence design system in DNA computing. In: Proceedings of the 2002 Congress on Evolutionary Computing, 2002, 73—78
16 Shin SY, Kim DM, Lee IH, et al. Evolutionary sequence generation for reliable DNA computing. In: Proceedings of the 2002 Congress on Evolutionary Computation, 2002, 79—84