ISA Transactions 48 (2009) 254–263

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

System identification: DNA computing approach

Ching-Huei Huang [a], Horn-Yong Jan [b], Chun-Liang Lin [a,*], Chia-Soon Lee [a]

[a] Department of Electrical Engineering, National Chung Hsing University, 250, Kuo Kuang Road, Taichung, 402, Taiwan, ROC

[b] Graduate Institute of Electrical and Communications Engineering, Feng Chia University, 100 Wenhwa Road, Taichung, 40724, Taiwan, ROC

ARTICLE INFO

Article history:
Received 20 June 2008
Received in revised form
15 December 2008
Accepted 26 January 2009
Available online 27 February 2009

Keywords:
DNA computing algorithm (DNACA)
Electron–ion interaction potential (EIIP)
Systems identification

ABSTRACT

A DNA computing algorithm (DNACA) with an electron–ion interaction potential (EIIP) decoding scheme is proposed to identify a class of transfer functions. The DNACA includes enzyme and virus operators which provide a highly modular, flexible, and accurate self-organizing structure environment. Simulation study based on De Jong's test functions show its superior performance when compared with the improved and standard genetic algorithms (GAs).

1. Introduction

Recently developed DNA computing algorithms (DNACAs) have inspired new methods that can simultaneously solve the parameter and structure optimization problems. The DNACAs based on the concept of bimolecular evolution was first developed by Adleman [1]. Maley [2] further detailed this kind of algorithm in terms of chemical processes and computer programming.

The operational features inherent in DNACAs make the algorithms in DNA computers which are over a billion times possible to implement more computationally efficient than the conventional computers. The massively parallel nature of DNA in those computers means that computing may be millions or billions of times beyond today's supercomputers (Forbes [3]).

In this paper, DNACA with an electron–ion interaction potential (EIIP) decoding scheme is proposed to identify a class of transfer functions (Cosic et al. [4]). Accuracy of the proposed approach with or without frameshift operators is verified first by numerical testing of De Jong's [5] test functions. With regard to system identification, the DNA strings are created to represent the transfer model in which codon is used to represent coefficients of the denominator and numerator polynomials. Using frameshift operators (enzyme and virus), the order and numeric values of the transfer model can be refined to fit the identified object. Verification of the simulated results show that the proposed method is quite well-performed, even for the systems with wide dynamic responses.

Applying DNA computing techniques carries distinct advantages and has great potential, as outlined here:

- Current molecular biotechnology is available for implementing DNA computing operators (Chen et al. [6], and Jan et al. [7]).
- Compared to conventional computers, certain operations can make DNA computing over a billion times more energy efficient. The massively parallel nature of DNA means that computing its sequence may be millions or billions of times beyond the reach of today's supercomputers [3].
- More capacity for data storage is available using DNA sequences. A gram of DNA, for example contains approximately 2.1×10^{21} bases, which has an information content of approximately 4.2×10^{21} bits, far more than the 200 petabyte storage of all the digital magnetic tape produced (Tsuboi et al. [8]).

This paper describes how a model selection problem can be solved by a synthesis of qualitative and quantitative techniques using a model structure and numeric computations. Accuracy can be verified by numerical analyses of De Jong's test functions through the proposed DNACA with or without frameshift operators. For system identification, an imitating model is constructed from DNA strings in which codons were used to represent the coefficients of the denominator and numerator polynomials. Using special frameshift operators, the order and the numeric values of the parameters can refine the properties of the modeling system more precisely.

2. Biological computation algorithms

A DNA strand consists of two complimentary strings in which four nucleotide base: adenine (A), cytosine (C), guanine (G), and thymine (T) are arranged in various combinations. Nucleotides are

* Corresponding author. Tel.: +886 4 22851549x708.
E-mail address: chunlin@dragon.nchu.edu.tw (C.-L. Lin).

0019-0578/$ – see front matter © 2009 ISA. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.isatra.2009.01.006

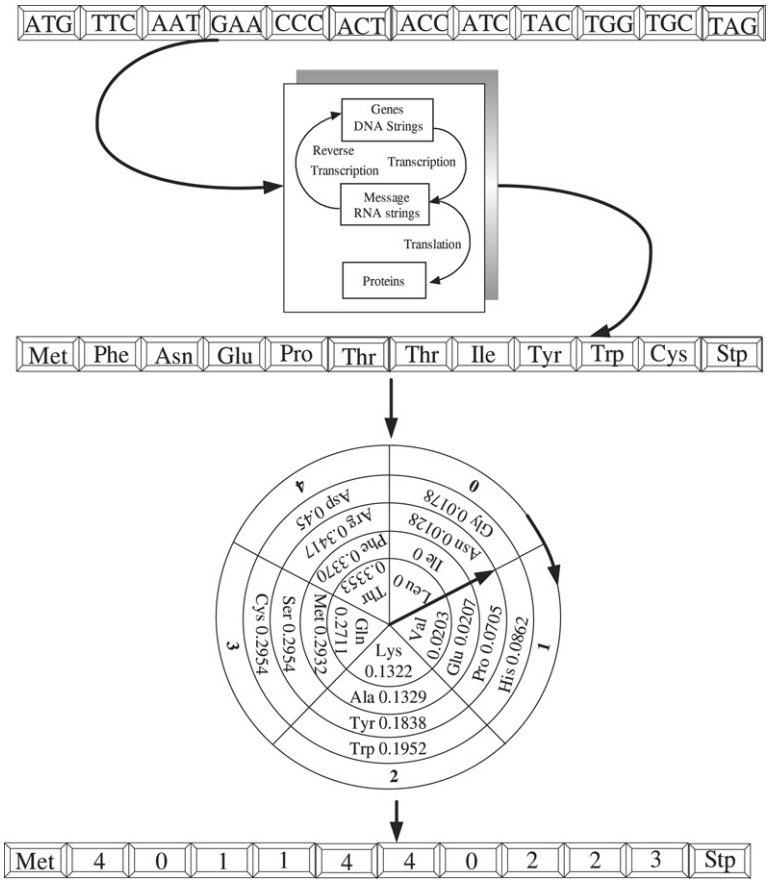

Fig. 1. Normalized EIIP wheel of amino acids.

paired along with two strings. For the process of computation, one DNA string is separated from the other through chemical process. As the other string is a perfect compliment, only one string is needed to further operations.

DNACAs consist of several operators in which a crossover is used to generate a new strand that will retain beneficial features from the parent generation. This exploits current genetic potential. If the population does not contain enough encoded information to solve a particular problem, none of the mixing strands can produce a satisfactory solution. For this reason, the mutation operator capable of spontaneously generating new strands provides a mechanism to maintain the population's diversity (Goldberg [9]).

In addition to the mutation operator, two frameshift operators are depicted in what follows.

2.1. Enzyme mutation

The enzyme mutation works to separate and connect two DNA substrings while removing a section of codon. The newly formed codon breaks away from the doped position and the reacted remainder of codon are joined together to form a new one. The direct consequence of this operation is that the resulting DNA string loses some information in the next generation.

2.2. Virus mutation

From the biological viewpoint, the major cause of disease is usually that a virus has intruded into the human body, modifying the original codon of DNA strings and reproducing the infected strings. Through this mechanism, the inserted DNA string makes an incursion into its parent's string to form a new one. The direct consequence of this operation is that the resulting DNA string gains extra information in the next generation.

3. Transfer function modeling based on DNACA

3.1. Configurable transfer functions

Consider a class of transfer function modeled as follows:

$$\widehat{C}_E(s) = \widehat{C}_C \frac{\prod_{i=1}^{n_n}(s+\widehat{C}_{N1\_i}) \prod_{i=1}^{m_n}(s^2+\widehat{C}_{N2\_i}s+\widehat{C}_{N3\_i})}{\prod_{i=1}^{n_d}(s+\widehat{C}_{D1\_i}) \prod_{i=1}^{m_d}(s^2+\widehat{C}_{D2\_i}s+\widehat{C}_{D3\_i})}. \quad (1)$$

Let $\tilde{n}_n = n_n + 2m_n$ and $\tilde{n}_d = n_d + 2m_d$ with $\tilde{n}_d \geq \tilde{n}_n$. $\widehat{C}_E(s)$ is said to be proper if $\tilde{n}_d \geq \tilde{n}_n$ and is improper if $\tilde{n}_d < \tilde{n}_n$.

In traditional system identification methodologies, the least mean square error scheme and some iterations are commonly adopted to build the transfer function model so that it ultimately mimics the relationship. However, the most desirable transfer function is not usually the one with minimal order. In the following sections, the DNACA is used to reconstruct the model and module of the estimated system.

3.2. DNA coding scheme

Biologists have discovered 20 amino acids encoded by 64 codons, which are expressed in the versatile combinations of {A, C, T, G}. As in biological DNA, a gene starts with the codon ATG and ends with the codon TAA, TAG or TGA.

Electron–ion interaction potential (EIIP) is applied here instead of binary coding system. In the presented method, each amino acid is represented by a specific number referred to as the unique electron–ion interaction potential and it is irrespective of its

Table 1
Translation of DNA codons.

Amino acid	Three-letter code		One-letter code	Code of amino acid	Electron–ion interaction potential values
Alanine	ALA	Ala(12)	A	GCT GCC GCA GCG	0.0373
Arginine	ARG	Arg(11)	R	CGT CGC CGA CGG AGA AGG	0.0959
Aspartic	ASP	Asp(17)	D	GAT GAC	0.1263
Asparagine	ASN	Asn(15)	N	AAT AAC	0.0036
Cysteine	CYS	Cys(19)	C	TGT TGC	0.0829
Glutamic	GLU	Glu(18)	E	GAA GAG	0.0058
Glutamine	GLN	Gln(14)	Q	CAA CAG	0.0761
Glycine	GLY	Gly(13)	G	GGT GGC GGA GGG	0.0050
Histidine	HIS	His(5)	H	CAT CAC	0.0242
Isoleucine	ILE	Ile(2)	I	ATT ATC ATA	0.0000
Leucine	LEU	Leu(9)	L	TTA TTG CTT CTC CTA CTA	0.0000
Lysine	LYS	Lys(16)	K	AAA AAG	0.0371
Mothionine	MET	Met(3)	M	ATG	0.0823
Phenylalanine	PHE	Phe(1)	F	TTT TTC	0.0946
Proline	PRO	Pro(7)	P	CCT CCC CCA CCG	0.0198
Serine	SER	Ser(10)	S	TCT TCC TCA TCG AGT AGC	0.0829
Threonine	THR	Thr(8)	T	ACT ACC ACA ACG	0.0941
Tryptophan	TRP	Trp(20)	W	TGG	0.0548
Tyrosine	TYR	Tyr(4)	Y	TAT TAC	0.0516
Valine	VAL	Val(6)	V	GTT GTC GTA GTG	0.0057

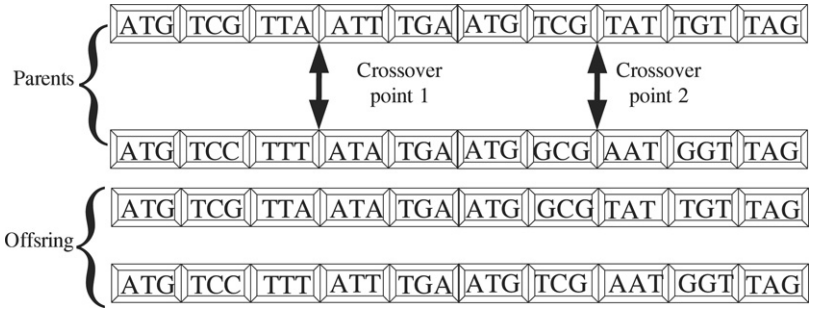

Fig. 2. Multi-point crossover operation.

position in a DNA string. Moreover, these numbers are essential to build a physical and mathematical model which interprets protein sequences linear information by using signal analysis methods.

The EIIP values for 20 amino acids and four nucleotides are summarized in Table 1. Normalization is then used to build an orthogonal based on these values, as shown in Fig. 1. It allows all amino acids to be divided into five groups based on their EIIP.

Each DNA string is encoded here as a vector in the following form:

$$\vec{S} = \left[\vec{S}_1 \quad \vec{S}_2 \quad \cdots \quad \vec{S}_\mu\right]. \tag{2}$$

The individuals are defined as in Box I, where $\vec{S}_{\mathrm{IND}}$ consists of the structure information $\vec{S}_{\mathrm{INF}}$ and the parameter information $\vec{S}_{\mathrm{PARA}}$. $\vec{S}_{\mathrm{INF}}$ represents the coding information for the system gain, denominator polynomial and numerator polynomial based on the combination of different numbers of amino acids. To fully specify the required information for characterizing the denominator or numerator polynomial, excluding lead and end codons, each needs 6 codon triplets denoted by $\vec{S}_{\mathrm{INF\_G,N,D(A)}} \sim \vec{S}_{\mathrm{INF\_G,N,D(F)}}$ which constitute the elements for the translation process. The sequence $\vec{S}_{\mathrm{PARA}}$ includes a constant term $S_{\mathrm{PARA\_G}}$, a numerator polynomial $\vec{S}_{\mathrm{PARA\_N}}$ and a denominator polynomial $\vec{S}_{\mathrm{PARA\_D}}$, based on various combinations of amino acids. S_{STA} and $S_{\mathrm{STP\_}S_s}$ with $S_s \triangleq$ INF, D, N were used to identify lead and end codons. Subscript notes (such as "f" or "s") are defined as the polynomials with the first- or second-order polynomial.

Normalization is performed based on the categorization of EIIP; the values for 20 amino acids and four nucleotides are summarized in Table 1. Accordingly, the decoded parameters are calculated, excluding lead and end codon, as follows

$$\frac{r\left(\sum_{i=1}^{l} \mathrm{DNA}^{(i)}_{AA\_EIIP} \times 5^{i-1}\right) - M}{N} \tag{3}$$

where the constants r, M and N control the range and resolution of the parameters; $\mathrm{DNA}^{(i)}_{AA\_EIIP}$ is the number of the ith amino acid of $\vec{S}_{\mathrm{PARA}}$; l is the length of the corresponding parameters.

3.3. Crossover operation

It causes an exchange of DNA information between two strings which have the same length via random decision in the mating pool and provide a mechanism to produce and match desirable qualities. Several string couples in the mating pool are randomly selected according to the crossover rate G_c and the multi-point scheme, as shown in Fig. 2.

3.3.1. Crossover information fragment

See Box II.

3.3.2. Crossover parameter fragment

See Box III, where L is length of the corresponding fragment, $\rho_{1,2}$ indicate the position of the crossover points. S_{PARA_i} and $i = N_f, D_s, N_f$ or D_s are defined as in Box I. $\mathrm{DNA}_{\mathrm{RES}}$ specifies the resolution for the corresponding term where each unit (codon) complies 3 nucleotides and τ is defined to specify the number of $\mathrm{DNA}_{\mathrm{RES}}$ for the parameter resolution.

$$\vec{S}_{\mathrm{IND}} = \left[\vec{S}_{\mathrm{INF}} \quad \vec{S}_{\mathrm{PARA}}\right], \quad \mathrm{IND} = 1, \ldots, \mu,$$
$$S_{\mathrm{STA}} \triangleq \mathrm{ATG},\ S_{\mathrm{STP}} \triangleq \{\mathrm{TAG}, \mathrm{TGA}, \mathrm{TAA}\},$$
$$\vec{S}_{\mathrm{INF}} = \left[\vec{S}_{\mathrm{INF\_G}} \quad \vec{S}_{\mathrm{INF\_D}} \quad \vec{S}_{\mathrm{INF\_N}}\right],$$
$$\vec{S}_{\mathrm{INF\_G}} = \left[S_{\mathrm{STA}} \quad S_{\mathrm{INF\_G}(1)} \quad S_{\mathrm{STP}} \quad S_{\mathrm{STP\_INF}}\right],$$
$$\vec{S}_{\mathrm{INF\_D}} = \left[S_{\mathrm{STA}} \quad S_{\mathrm{INF\_D}(1)} \quad \cdots \quad S_{\mathrm{INF\_D}(7)} \quad S_{\mathrm{STP}} \quad S_{\mathrm{STP\_INF}}\right],$$
$$\vec{S}_{\mathrm{INF\_N}} = \left[S_{\mathrm{STA}} \quad S_{\mathrm{INF\_N}(1)} \quad \cdots \quad S_{\mathrm{INF\_N}(7)} \quad S_{\mathrm{STP}} \quad S_{\mathrm{STP\_INF}}\right],$$
$$S_{\mathrm{STP\_INF}} \triangleq \mathrm{TGATGA},$$
$$\vec{S}_{\mathrm{PARA}} = \left[S_{\mathrm{PARA\_G}} \quad \vec{S}_{\mathrm{PARA\_D}} \quad \vec{S}_{\mathrm{PARA\_N}}\right],$$
$$S_{\mathrm{PARA\_G}} = \left[S_{\mathrm{STA}} S_{\widehat{C}_C} S_{\mathrm{STP}} S_{\mathrm{STP\_G}}\right],$$
$$\vec{S}_{\mathrm{PARA\_D}} = \left[\vec{S}_{\mathrm{PARA\_D_f}} \quad \vec{S}_{\mathrm{PARA\_D_s}}\right] = \left[S_{\mathrm{STA}} S_{\widehat{C}_{N1\_1}} S_{\mathrm{STP}} S_{\mathrm{STA}} S_{\widehat{C}_{N1\_2}} \cdots S_{\mathrm{STA}} S_{\widehat{C}_{N1\_n_n}} S_{\mathrm{STP}} S_{\mathrm{STA}} S_{\widehat{C}_{N2\_1}} S_{\mathrm{STP}} S_{\mathrm{STA}} S_{\widehat{C}_{N3\_1}} \cdots S_{\widehat{C}_{N3\_m_n}} S_{\mathrm{STP}} S_{\mathrm{STP\_N}}\right],$$
$$\vec{S}_{\mathrm{PARA\_N}} = \left[\vec{S}_{\mathrm{PARA\_N_f}} \quad \vec{S}_{\mathrm{PARA\_N_s}}\right] = \left[S_{\mathrm{STA}} S_{\widehat{C}_{D1\_1}} S_{\mathrm{STP}} S_{\mathrm{STA}} S_{\widehat{C}_{D1\_2}} \cdots S_{\mathrm{STA}} S_{\widehat{C}_{D1\_n_d}} S_{\mathrm{STP}} S_{\mathrm{STA}} S_{\widehat{C}_{D2\_1}} S_{\mathrm{STP}} S_{\mathrm{STA}} S_{\widehat{C}_{D3\_1}} \cdots S_{\widehat{C}_{D3\_m_d}} S_{\mathrm{STP}} S_{\mathrm{STP\_D}}\right],$$
$$S_{\mathrm{STP\_G}} \triangleq \mathrm{TGA},\ S_{\mathrm{STP\_D}} \triangleq \mathrm{TAA},\ S_{\mathrm{STP\_N}} \triangleq \mathrm{TAG}$$

Box I.

$$\begin{Bmatrix} \vec{S}^t_{C(\mathrm{IND})} \\ \vec{S}^t_{C(\mathrm{IND}+1)} \end{Bmatrix} = O_{\{G_c\}}\left(\begin{Bmatrix} \vec{S}^t_{(\mathrm{IND})} \\ \vec{S}^t_{(\mathrm{IND}+1)} \end{Bmatrix}\right), \quad \forall\, \mathrm{IND} \in \{1, 2, \ldots, \mu - 1\}$$
$$\Rightarrow \left\{ \begin{array}{c} \left[S_{\mathrm{INF}\_i(\mathrm{IND}+1),2}, S_{\mathrm{INF}\_i(\mathrm{IND}+1),3}, \ldots, S_{\mathrm{INF}\_i(\mathrm{IND}+1),\rho_1}, S_{\mathrm{INF}\_i(\mathrm{IND}),(\rho_1+1)}, \right. \\ \left. \ldots, S_{\mathrm{INF}\_i(\mathrm{IND}),\rho_2}, S_{\mathrm{INF}\_i(\mathrm{IND}+1),(\rho_2+1)}, \ldots, S_{\mathrm{INF}\_i(\mathrm{IND}+1),L}\right] \\ \left[S_{\mathrm{INF}\_i(\mathrm{IND}),2}, S_{\mathrm{INF}\_i(\mathrm{IND}),3}, \ldots, S_{\mathrm{INF}\_i(\mathrm{IND}),\rho_1}, S_{\mathrm{INF}\_i(\mathrm{IND}+1),(\rho_1+1)}, \right. \\ \left. \ldots, S_{\mathrm{INF}\_i(\mathrm{IND}+1),\rho_2}, S_{\mathrm{INF}\_i(\mathrm{IND}),(\rho_2+1)}, \ldots, S_{\mathrm{INF}\_i(\mathrm{IND}),L}\right] \end{array} \right\}, \quad 1 \le \rho_1 < \rho_2 \le L,$$
$$\begin{cases} L = \mathrm{DNA}_{\mathrm{RES}}, & \text{if } i = G \\ L = 7 \cdot \mathrm{DNA}_{\mathrm{RES}}, & \text{if } i = N \text{ or } D \end{cases}$$

Box II.

$$\Rightarrow \left\{ \begin{array}{c} \left[S_{\mathrm{PARA}\_i(\mathrm{IND}+1),1}, S_{\mathrm{PARA}\_i(\mathrm{IND}+1),2}, \ldots, S_{\mathrm{PARA}\_i(\mathrm{IND}+1),\rho_1}, S_{\mathrm{PARA}\_i(\mathrm{IND}),(\rho_1+1)}, \right. \\ \left. \ldots, S_{\mathrm{PARA}\_i(\mathrm{IND}),\rho_2}, S_{\mathrm{PARA}\_i(\mathrm{IND}+1),(\rho_2+1)}, \ldots, S_{\mathrm{PARA}\_i(\mathrm{IND}+1),L}\right] \\ \left[S_{\mathrm{PARA}\_i(\mathrm{IND}),1}, S_{\mathrm{PARA}\_i(\mathrm{IND}),2}, \ldots, S_{\mathrm{PARA}\_i(\mathrm{IND}),\rho_1}, S_{\mathrm{PARA}\_i(\mathrm{IND}+1),(\rho_1+1)}, \right. \\ \left. \ldots, S_{\mathrm{PARA}\_i(\mathrm{IND}+1),\rho_2}, S_{\mathrm{PARA}\_i(\mathrm{IND}),(\rho_2+1)}, \ldots, S_{\mathrm{PARA}\_i(\mathrm{IND}),L}\right] \end{array} \right\}, \quad 1 \le \rho_1 < \rho_2 \le L,$$
$$\begin{cases} L = \tau \mathrm{DNA}_{\mathrm{RES}}, & \text{if } i = G \\ L = \tau \mathrm{DNA}_{\mathrm{RES}} S_{\mathrm{INF}\_j,4} \quad \text{and} \quad j = N \text{ or } D, & \text{if } i = N_f \text{ or } D_f \\ L = \tau \mathrm{DNA}_{\mathrm{RES}} S_{\mathrm{INF}\_j,5} \quad \text{and} \quad j = N \text{ or } D, & \text{if } i = N_s \text{ or } D_s \end{cases}$$

Box III.

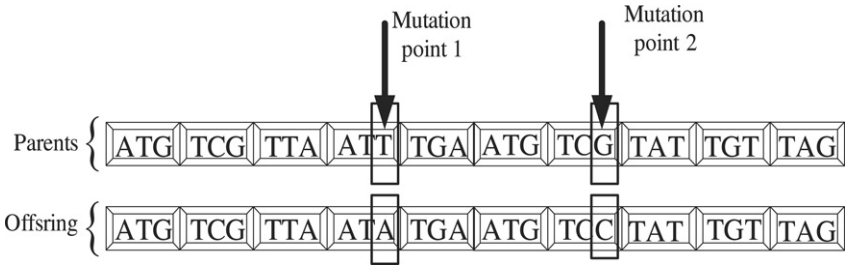

Fig. 3. Multi-point mutation operation.

3.4. Mutation operation

A multi-point mutation is applied according to the mutation rate G_m and the principle of codon exchange A ↔ T and G ↔ C to ensure that all points in the search space could be ultimately reached. A mutation would simultaneously cause data alternation on the two fragments, in Fig. 3.

3.4.1. Mutation information fragment

See Box IV.

3.4.2. Mutation parameter fragment

See Box V,

3.5. Enzyme and virus mutation

Enzyme and virus operators are randomly applied to alter the structure and parameters of the DNA strings during translation and evolution. Through the end codon, the segment corresponding to the first- and second-order polynomials that being selected for insertion or deletion of codon can be identified. Enzyme and virus operators also apply to the parameter fragment to generate a variant resolution of the parameters. The two operators work by

$$\vec{S}^t_{m(\mathrm{IND})} = O_{\{G_m\}}(\vec{S}^t_{c(\mathrm{IND})}), \quad \forall\, \mathrm{IND} \in \{1, \ldots, \mu - 1\}$$

$$\vec{S}^t_{m(\mathrm{INF\_}i(IND),k)} = \begin{cases} \vec{S}^t_{c(\mathrm{INF\_}i(\mathrm{IND}),k)}, & k \in \{1, \ldots, p-1, p+1, \ldots, L\} \\ \vec{\exists}^t_{c(\mathrm{INF\_}i(\mathrm{IND}),k)}, & k = p, \end{cases} \qquad 1 \le p < L.$$

Box IV.

$$\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),k)} = \begin{cases} \vec{S}^t_{c(\mathrm{PARA\_}i(\mathrm{IND}),k)}, & k \in \{1, \ldots, p-1, p+1, \ldots, L\} \\ \vec{\exists}^t_{c(\mathrm{PARA\_}i(\mathrm{IND}),k)}, & k = p, \end{cases} \qquad 1 \le p < L$$

where p indicates the location of the mutation point.

Box V.

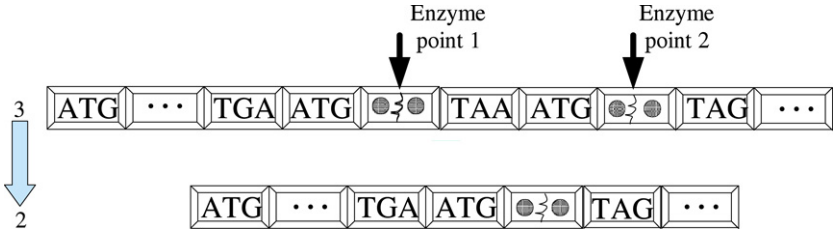

Fig. 4. Enzyme mutation operation.

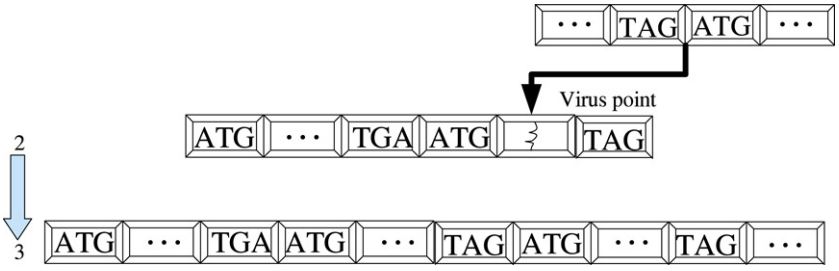

Fig. 5. Virus mutation operation.

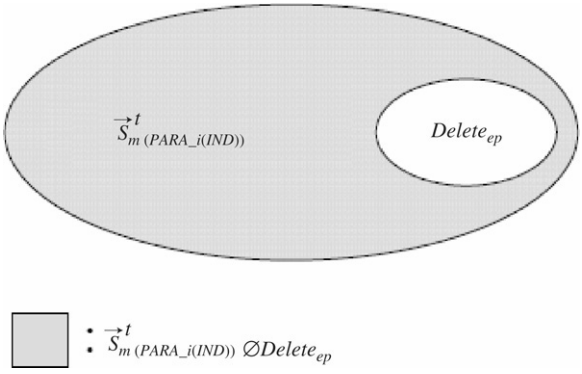

Fig. 6a. Illustrations of enzyme mutation operations.

randomly altering n consecutive string units with each string's unit representing a control coefficient based on the deletion rate G_e and the insertion rate G_v, respectively, as shown in Figs. 4 and 5.

3.5.1. Enzyme mutation

The enzyme operation works as a bit deletion mutation, it can be divided into separation and connection. In the former, the reacting molecules go through chemical changes rapidly. The newly formed molecules break away from the doped position, as shown in Fig. 6a.

$$\{\vec{S}^t_{e(\mathrm{PARA\_}i(\mathrm{IND}))}\} \triangleq O_{\{G_e\}}\{\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}))}\}, \quad \forall\, \mathrm{IND} \in \{1, 2, \ldots, \mu_e\}$$
$$\triangleq \left\{\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}))} \emptyset Delete_{ep}\right\},$$

where

$$\{S^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),2)}, S^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),3)}, \ldots, S^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),L-3)}\}, \tag{4}$$

$$Delete_{ep} = \vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),\rho^e_j)}, \quad 1 < \rho^e_i < L,$$

$$\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),\rho^e_j)} \in X_k$$

and $X_k \in \vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}))}$, $X_k = (x_1x_2x_3), x_i \in \{\mathrm{A, G, T, C}\}$

$\emptyset$ means the excision of enzyme mutation operations.

where the length L is defined as in Box I.

3.5.2. Virus mutation

From a biological viewpoint, one of the major causes of disease is that a virus intrudes into the human body, it works as a bit insertion mutation, as shown in Fig. 6b.

$$\{\vec{S}^t_{v(\mathrm{PARA\_}i(\mathrm{IND}))}\} \triangleq O_{\{G_v\}}\{\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}))}\}, \quad \forall \mathrm{IND} \in \{1, 2, 3, \ldots, \mu_v\}$$
$$\triangleq \left\{\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}))} \oplus Insert_{vp}\right\}$$

where

$$\{S^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),2)}, \ldots, Insert_{vp}, \ldots, S^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),L+3)}\}, \tag{5}$$

$$Insert_{vp} = \vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),\rho^v_i)}, \quad 1 < \rho^v_i < L,$$

$$\vec{S}^t_{m(\mathrm{PARA\_}i(\mathrm{IND}),\rho^v_i)} \in X_k, X_k = (x_1x_2x_3), x_i \in \{\mathrm{A, G, T, C}\}$$

$\oplus$ means the combination of the virus mutation operations

where the length L is defined as in Box I.

Table 2
Results searched by the proposed DNACA, the standard GA and the improved GA for De Jong's function tests.

De Jong's test functions (converging generation)	Proposed DNACA	Standard GA	Improved GA
$D_1(x)$	1.0000(25)	1.0000(225)	1.0000(100)
$D_2(x)$	1.0000(20)	0.6393(>450)	0.9707(>045)
$D_3(x)$	0.9995(40)	0.8037(>450)	0.8349(>450)
$D_4(x)$	1.0000(25)	1.0000(200)	1.0000(>450)
$D_5(x)$	1.0000(45)	0.7297(>500)	1.0000(75)

Table 3
Comparison of DNACA efficiency with/without frameshift operators.

Test functions	D_4 with two parameters		D_5 with three parameters	
	Condition			
	With frameshift operators	Without frameshift operators	With frameshift operators	Without frameshift operators
Fitness value	1.000	0.991	1.000	0.6465
Converging generation	20	50	45	45
Length	8, 3 codons	6, 6 codons	1, 2, 2 codons	6, 6, 6 codons
DNA string and decimal values of parameters	CATTTCTCAGTT AAATATGAAGTC (−31.4014) GCACAGTCT (31.6239)	ACGGCGCCG GCCCACGAA (−31.5653) AGGGCCAAA GCGCAGATG (31.3896)	TTA(0) TTGGGA(0) TACTGG(0)	TTTTTTAGACCT TATAAG(−0.0413) ACCCAGAACTAT TGGTGG(−0.0282) TATTGGGAAGCT GCAGCA(−0.0164)

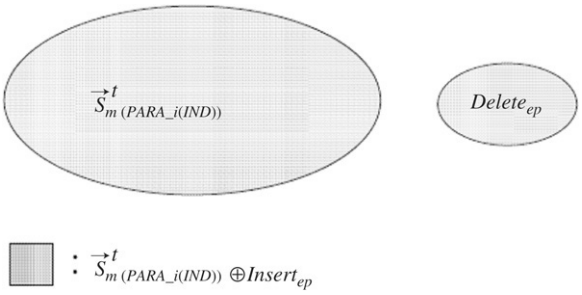

Fig. 6b. Illustrations of virus mutation operations.

Succeeding to one of the frameshift mutations there is a translation process to recount the length of the parameter fragment and examine the proper condition for the information fragment that ensures not only a realizable controller but maintains consistency of the two fragments. After each operation, a translation process is applied so that the corresponding parameter fragment is updated accordingly.

3.6. Objective function

The objective function for the current problem is defined as follows

$$J = \sum_i w_i J_i \tag{6}$$

where w_i is the weighting factor for the corresponding term, three objective functions are chosen:

$$J_1(\vec{S}_{\mathrm{IND}}) : G_e^{Low} = \left| \frac{Mag(\widehat{P}_E(j\omega_{Lf}))}{10^{-\exp(\log(\omega_{Lf}))}} - \frac{Mag(P_0(j\omega_{Lf}))}{10^{-\exp(\log(\omega_{Lf}))}} \right| \tag{7}$$

$$J_2(\vec{S}_{\mathrm{IND}}) : M_e = \sum_{i=1}^{f_t} \left| \ln(Mag(\widehat{P}_E(j\omega_i))) - \ln(Mag(P_0(j\omega_i))) \right| \tag{8}$$

$$J_3(\vec{S}_{\mathrm{IND}}) : PA_e = \sum_{i=1}^{f_t} \left| phase(\widehat{P}_E(j\omega_i)) - phase(P_0(j\omega_i)) \right| \tag{9}$$

and ω_{Lf} represents the lowest frequency; $Mag(\cdot)$ and $phase(\cdot)$ denote, respectively, the magnitude and phase of the frequency response obtained at ω_i; f_t is the total sampling number over the testing frequency interval. Here, J_1 denotes the magnitude of gain error at the lowest frequency; whereas J_2 and J_3 calculate, respectively, the overall errors of the magnitude and phase over the entire spectrum.

Although a complicated transfer function model generally enables superior performance, a simpler structure with at least acceptable performance is more meaningful from the practical viewpoint. Therefore, the objective function considered should contain not only system performance indices but also structure information. The general form of the performance index is defined as follows

$$F' = (1-\beta)J_{\mathrm{ind}} + \beta g_{\mathrm{ind}} \tag{10}$$

where

$$J_{\mathrm{ind}}(\vec{S}_{\mathrm{IND}}) = \sum_{i=1}^{3} \alpha_i J_i(\vec{S}_{\mathrm{IND}}) \tag{11}$$

$$g_{\mathrm{ind}}(\vec{S}_{\mathrm{IND}}) = \frac{\tilde{n}_d^{\mathrm{IND}}}{\max\limits_{1\le \mathrm{IND}\le \mu} \tilde{n}_d^{\mathrm{IND}}} \tag{12}$$

and F' is made up of the individual index J_i with α_i representing the corresponding weighting factor and β representing the desired emphasis on control complexity; $\tilde{n}_d^{\mathrm{IND}}$ is the orders of the INDth's denominator and it can be derived by counting the numbers of the leading and ending codon except for the first leading and final ending codon. The flowchart of DNACA for the optimal solution searching is illustrated in Fig. 7.

4. Cases study

Case 1: For De Jong's functions testing

To examine applicability and efficiency of the proposed DNACA, the following De Jong's test functions (De Jong's [5], Hanaki et al. [10], Frank et al. [11]) are considered:

$$D_1 = \sum_{i=1}^{3} x_i^2, \quad -5.12 \le x_i \le 5.12 \tag{13}$$

where D_1 is a simple 3D parabola with a spherical constant-cost contour.

$$D_2 = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2, \quad -2.048 \le x_i \le 2.048 \tag{14}$$

Fig. 7. Flowchart of DNACA algorithm for the optimal solution searching.

where D_2 is a typical test function to deal with the optimization. The point (1,1) is a minimum of zero—it is difficult to perceive because a deep parabolic valley is along with the relation $x_2 = x_1^2$.

$$D_3 = \sum_{i=1}^{3} i x_i^4 + \text{Gauss}(0, 1), \quad -1.28 \le x_i \le 1.28 \tag{15}$$

where D_3 is a continuous, convex, unimodal, and 3D quadric which function with zero-mean Gaussian noise.

$$D_4 = 0.002 - \sum_{i=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6},$$
$$-65.356 \le x_i \le 65.356 \tag{16}$$

where D_4 is a multi-modal function; it has 25 local minimums lying approximately at the points a_{ij}, $i = 1, 2$ and $j = 1, \ldots, 25$.

$$D_5 = \sum_{i=1}^{3} [x_i^2 - 10\cos(2\pi x_i) + 10], \quad -5.12 \le x_i \le 5.12 \tag{17}$$

where D_5 is a generalized Rastrigin's function. It has multiple local optimum that can be used to examine the efficiency of the optimization algorithms.

The corresponding fitness functions F_D^1 for $D_{1,2,3,5}$ and F_D^2 for D_4 are given, respectively, as

$$F_D^1 = \frac{1}{1 + D_i(x)}, \quad i = 1, 2, 3, 5; \qquad F_D^2 = \frac{1}{(|1 - D_4| + 1)}. \tag{18}$$

For each test function, the iteration and population numbers are set to be 50 and 10 respectively. The probability rates of crossover and mutation are 0.8 and 0.2, respectively. Both of the probability rates of enzyme and virus are 0.4. The results of the solution are searched by using the proposed DNACA as summarized in Table 2. Taking D_4 as the example, the efficiency of DNACA with/without frameshift operators is illustrated as in Table 3. Convergence of the fitness values are shown in Figs. 8 and 9 is displayed the values of D_4 and D_5. It is seen that the DNACA can produce not only the accurate solutions, it simultaneously prevents the redundant parts of candidate strings from being repeated in the evolutionary process. These results demonstrate superior efficiency and performance of the present algorithm while comparing to other approaches.

We proceed to show how the DNACA can be used to identify the following system model:

$$P_0 = \frac{1.038\left(\frac{s}{300} + 1\right)\left(\frac{s^2}{0.0911^2} + \frac{2\times 1.17s}{0.0911} + 1\right)\left(\frac{s^2}{0.7587^2} + \frac{2\times 1.129s}{0.7587} + 1\right)}{s\left(\frac{s}{4.012} + 1\right)\left(\frac{s^2}{0.1068^2} + \frac{2\times 2.712s}{0.1068} + 1\right)\left(\frac{s^2}{2.168^2} + \frac{2\times 1.59s}{2.168} + 1\right)}. \tag{19}$$