DNA computing based RNA genetic algorithm with applications in parameter estimation of chemical engineering processes

Jili Tao, Ning Wang *

National Laboratory of Industrial Control Technology, Institute of Advanced Process Control, Zhejiang University, Hangzhou 310027, PR China

Received 15 December 2005; received in revised form 29 January 2007; accepted 29 January 2007

Available online 6 February 2007

Abstract

Based on RNA genetic operations and DNA sequence model under selection and mutation, an electronic RNA genetic algorithm (RNA-GA) with improved crossover and mutation operator is proposed. The proposed algorithm can be implemented on real biochemical reaction after simple transition, thus, the brute force method of DNA computing can be broken. The convergence analysis of the proposed algorithm shows that RNA-GA with elitist strategy can converge in probability to the global optimum. Comparisons of RNA-GA with standard genetic algorithm (SGA) for typical test functions show the advantages and efficiency of the proposed algorithm. As illustrations, the RNA-GA is implemented on parameter estimation of a heavy oil thermal cracking 3-lumping model and a fluid catalytic cracking unit (FCCU) main fractionator. In both cases, it is shown that the methodology is effective in parameter estimation of chemical processes.

© 2007 Published by Elsevier Ltd.

Keywords: RNA; DNA computing; Standard genetic algorithm (SGA); Convergence analysis; Parameter estimation

1. Introduction

Since Adleman (1994) first introduced DNA computing by solving a computationally hard problem of the directed Hamiltonian path problem, many groups have worked on different NP hard problems with fewer variables, such as maximal clique problem (Ouyang et al., 1997), 3-SAT problems (Braich et al., 2002), DES deciphering problem (Boneh, Dunworth, & Lipton, 1995), traveling salesman problems (Lee et al., 2004), etc. Conventionally, Adleman-style DNA computing consists of three major steps: (1) generate a data pool of DNA molecules that represent all possible solutions to the studied problem, (2) utilize a series of biology laboratory techniques to exclude the DNA strands that do not match the logic constraints of the problem, (3) collect the surviving DNA molecules for the answer readout process. According to the above steps, DNA computing requires that the size of initial data pool increase exponentially with the number of variables in calculation, so this kind of DNA computing method is a brute force method. Genuinely, the difficulty is not the absence of correct strands after computing, but the presence of vast contaminating DNA. In order to break the barrier of this brute-force method and implement the DNA operations with an existed digital computer, various improved DNA computing methods and electronic DNA computing algorithms have been studied. Yang and Yang (2005) modified a well-known sticker model to build solution sequences in parts satisfying one clause in a step, and eventually solved the whole Boolean formula after a number of steps. Yamamura et al. (2002) proposed a local search method based on DNA concentration computing to solve the shortest path problem. Because laboratory experiments in DNA computing are highly difficult, inefficient, un-scalable and expensive compared to conventional computing standard, most of improved DNA computing methods are carried out theoretically. Hence, Garzon et al. (1999) described an electronic DNA (EDNA) to simulate a virtual test tube with digital computer and reproduced Adleman’s experiment. Hartemink et al. (1999) simulated biological reactions of DNA computing and implemented a simulator called CYBERCYCLER. Ouyang and co-workers proposed a genetic DNA computing algorithm to solve the maximal clique problem, which was possible to get a solution...
Nomenclature

\[ \begin{align*} 
  a & \quad \text{speed of the change of mutation probability} \\
  a_1 & \quad \text{ultimate mutation probability of } P_{mhl} \text{ and the initial value of } p_{ml} \\
  b_1 & \quad \text{range of transmutability} \\
  c_{ij} & \quad \text{probability of changing sequence } i \text{ into sequence } j \text{ by crossover operator} \\
  C & \quad \text{probability matrix of crossover} \\
  CM_i & \quad \text{ith manipulated variable} \\
  CV_i & \quad \text{ith controlled variable} \\
  \hat{C}V_i & \quad \text{output of the model to be optimized} \\
  \bar{E} & \quad \text{average evolution number of the optimization problem} \\
  E_{\text{min}} & \quad \text{minimum evolution number of the optimization problem} \\
  E_{\text{max}} & \quad \text{maximum evolution number of the optimization problem} \\
  F & \quad \text{average value of the optimization problem} \\
  F_{\text{min}} & \quad \text{minimum of the optimization problem} \\
  F_{\text{max}} & \quad \text{maximum of the optimization problem} \\
  g & \quad \text{evolution generation} \\
  g_0 & \quad \text{generation where great change of mutation probability occurs} \\
  H & \quad \text{individual Hamming distance} \\
  l & \quad \text{length of substring in an individual} \\
  L & \quad \text{length of individual, } L = n \times l \\
  m_{jk} & \quad \text{probability of sequence turning } j \text{ into sequence } k \text{ by mutation} \\
  M & \quad \text{probability matrix of mutation} \\
  n & \quad \text{number of substring in an individual} \\
  N & \quad \text{population size} \\
  N_s & \quad \text{number of samples} \\
  p_c & \quad \text{probability of crossover} \\
  p_m & \quad \text{probability of mutation} \\
  P_{mhl} & \quad \text{probability of mutation in high bit position} \\
  P_{ml} & \quad \text{probability of mutation in low bit position} \\
  p(t) & \quad \text{probability of changing } x \text{ into } y \\
  P & \quad \text{transition probability matrix} \\
  R & \quad \text{RNA sequence} \\
  R_i & \quad \text{RNA subsequence, } i = 1, 2, \ldots, 5 \\
  R' & \quad \text{new RNA sequence after operation} \\
  S & \quad \text{probability matrix of selection} \\
  S_{klq} & \quad \text{probability of changing sequence } k \text{ into sequence } q \text{ by selection operator} \\
  u(t) & \quad \text{system input vector} \\
  x & \quad \text{individual in the population} \\
  x_i & \quad \text{variables in global optimization problem } i = 1, 2, \ldots, n \\
  x_L & \quad \text{dependent variable a heavy oil thermal cracking three lumping model} \\
  y & \quad \text{individual differing from } x \\
  y(t) & \quad \text{system output} \\
  \hat{y}(t) & \quad \text{model output} \\
  z, T & \quad \text{independent variables of a heavy oil thermal cracking three lumping model} \\
 \end{align*} \]

\[ \begin{align*} 
  \Omega & \quad \text{sequence set of RNA} \\
  \Omega^N & \quad \text{all possible population in } \Omega \\
  \theta & \quad \text{parameter vector to be estimated} \\
  (\cdot) & \quad \text{reduce (an exact figure) to the nearest integer} \\
 \end{align*} \]

from a very small initial data pool and avoided enumerating all candidate solutions (Li, Fang, & Ouyang, 2004).

Genetic algorithm (GA), presented by Holland (1975), is a parallel, global optimization method with the search strategy partly similar to DNA computing. It may be one of the possible ways to be adopted to break the barrier of DNA computing and to make it practical as the problem size scales up. However, the double helix structure of DNA molecular is not suitable to be combined with the chromosome of GA.

Recently, RNA computing has been developed based on DNA computing. Cukras, Faulhammer, Lipton, and Landweber (1999) developed the theory of RNA computing and proposed a destructive algorithm to solve the knight problem using only biological molecules and enzymes. Lipton suggested that DNA be replaced by RNA in DNA computing (Faulhammer et al., 2000), and Li and Xu (2003b) summarized all possible operations of RNA sequences, such as elongation operation, deletion operation, absent operation, insertion operation, translocation operation, transformation operation and permutation operation, etc. By introducing the complementary oligonucleotides of DNA molecules, RNA strands obtain DNA genetic information. The unique single chain structure and various operations of RNA strands make it easy to combine with SGA. Furthermore, the genealogical processes have been the subject of much research in recent years. Neuhauser and Krone (1997) introduced several models including DNA sequence models to study the genealogy of a random sample of genes, which are taken from a large haploid population that evolved according to random reproduction with selection and mutation. Enlightened by the DNA sequence model and its distribution rules, a digital RNA-GA is proposed and its convergence is analyzed. The algorithm used in this work is essentially an improvement of SGA. Both the crossover operator based on RNA operations and the mutation operator based on DNA sequence model are introduced to the proposed algorithm, which increase the genetic diversity in the population. Simulation studies on several test functions show the efficiency of the RNA-GA. Parameter estimation for process modeling is a very important step in the control, diagnosis and optimization of the process system. The parameter estimation for chemical process modeling is especially difficult because of its non-linear and complicated characteristics. In Song et al. (2003), there are totally 8 parameters to be estimated in a heavy oil thermal cracking 3-lumping model, the traditional parameter estimation method, such as least square method, cannot be used in the chemical processes because of its non-linearity. Similarly, the parameter estimation of a FCCU main fractionator (Zhong & Wang, 1998) with variable coupling is difficult for the traditional parameter estimation. In this paper, both cases are implemented successfully by RNA-GA. Thus, this work focuses
on two aspects: (1) the development of the RNA-GA operators and the convergence analysis of RNA-GA and (2) its usage for test functions and parameter estimation of chemical processes.

2. RNA-GA based on DNA computing

2.1. Digital encoding of RNA sequence

The type space for a RNA sequence is $E = \{A, U, G, C\}^{l}$, i.e., sequences of length $L$, where four nucleotide bases Adenine(A), Uracil(U), Guanine(G), Cytosine(C) are utilized to encode the solution of the given problem in RNA computing. However, such RNA sequence cannot be processed by digital computer. Since the binary digital coding (00, 01, 10, 11) can represent the characteristics of RNA nucleotide bases, such as structure, function group, complementary relationship and the number of hydrogen bond connection, it is adopted to encode the 4 RNA nucleotide bases. There exist totally $P_4^4 = 24$ possible coding patterns, and the format (0123/CUAG) obtained by the molecular weight of the nucleotide bases is the best encoding pattern (Li & Xu, 2003a). For the convenience of the mathematical and logical operations, the first bit in the above encoding digit is defined as the structure bit, the second bit as the function bit, i.e., $1 \times$ delegates the purine base, $0 \times$ delegates the pyridine base, $\times 0$ represents the keto group, $\times 1$ represents the amido group, where $\times$ is 0 or 1 (Li & Xu, 2003b). Hence, Cytosine(C) accords with 00, Uracil(U) with 01, Adenine(A) with 10, Guanine(G) with 11. All of the following discussions are based on this digital encoding.

2.2. Operations of RNA molecule

There are various operations of RNA sequences (Li & Xu, 2003b). However, the genetic operations change the length of RNA sequence thus are not introduced since the length of individual chromosomes keeps invariable in SGA. The genetic operations which can be applied to RNA-GA are listed as follows.

There are three main operations on a single RNA sequence: translocation, transformation and permutation.

1. **Translocation operator**: Make subsequence of RNA sequence transfer to the new location. For example, suppose that the original RNA sequence is $R = R_2R_3R_4R_5R_1$, where $R_i$ ($i = 1, 2, \ldots, 5$) is the subsequence of RNA sequence ($R$), which is composed of four nucleotide bases (0123/CUAG), then the new sequence after translocation becomes $R' = R_3R_2R_4R_5R_1$.

2. **Transformation operator**: Let two segments of RNA sequence exchange their locations. For example, the sequence $R$ becomes $R' = R_3R_2R_4R_5R_1$, after exchanging $R_4$ with $R_2$.

3. **Permutation operator**: One subsequence of RNA sequence is permuted by the other subsequence. For example, when $R_2$ subsequence from the same or other RNA sequence is selected to replace $R_2$, the new sequence is $R' = R_3R_4R_5R_2R_1$.

Because there exist four elements in RNA sequence, the mutation of nucleotide base is relatively complex, which is defined as follows.

1. **Reversal operator**: The function bit of RNA nucleotide base is inverted while its structure bit keeps invariable, i.e. the premier digit of RNA encoding is reversed. There are four cases in total: $C \leftrightarrow G$, $U \leftrightarrow A$, i.e., $0 \leftrightarrow 2$, $1 \leftrightarrow 3$.

2. **Transition operator**: Contrary to reversal operator, the function bit of RNA nucleotide base keeps invariable while its structure bit is inverted. There also exist four cases: $C \leftrightarrow U$, $A \leftrightarrow G$, i.e., $0 \leftrightarrow 1$, $2 \leftrightarrow 3$.

3. **Exchange operator**: Both structure bit and function bit are transformed, i.e., the premier bit and the second bit of RNA nucleotide base are reversed, which yields the complementary sequence of the given RNA sequence. The four cases are: $A \leftrightarrow U$, $C \leftrightarrow G$, i.e., $2 \leftrightarrow 1$, $0 \leftrightarrow 3$.

2.3. The standard genetic algorithm (SGA)

Without loss of generality, the global optimization problem of the form (1) to (2) often arises as:

Minimize $f(x_1, x_2, \ldots, x_n)$ \hspace{1cm} (1)

Subject to $x_{\min_i} \leq x_i \leq x_{\max_i}$, $g(x_j) \leq 0; \hspace{1cm} i = 1, \ldots, n, 1 \leq j \leq n \hspace{1cm} (2)$

where $x_i$ ($i = 1, \ldots, n$) is the design variable, $g(x_j)$ ($1 \leq j \leq n$) the inequality constraint on some or all the design variables, and $[x_{\min_i}, x_{\max_i}]$ is the feasible range of the design variables. Through transformation, the maximization problem can also be in the form of (1) and (2).

GA is a global optimization method based on the principle of survival of the fittest. It is described as follows: each variable $x_i$ of the global optimization problem (1) and (2) is represented as a binary substring of a specified length ($l$), the lower limit of the search space $x_{\min_i}$ is thus represented by the decoded integer 0, while the upper limit $x_{\max_i}$ is represented by the decoded integer $2^l - 1$. A point in the $n$-dimensional search space consists of $n$ decision variables, $x_i, i = 1, \ldots, n$, which are represented as a chromosome of length $L = n \times l$. The processes of function evolution, selection, crossover, and mutation are described for SGA in the following steps:

Step 1: Generate the code for $N$ chromosomes randomly in the search space, where $N$ is the population size.

Step 2: Decode and compute the performance $f$ of each of the individual.

Step 3: Select the chromosomes (parents) to generate new chromosomes (children) of the next generation according to selection operator. Two commonly used methods are proportionate and tournament selection, and proportionate selection is adopted in this paper.

Step 4: Select a point randomly in the range $n \times l$, the bit length of each chromosome, and exchange the codes of the pairs of parents selected in step 3. Repeat this for all the
Step 5: For effect mutation, reverse the values of the bits (replace 0 with 1 and 1 with 0) of \( p_m \times l \times n \times N \) randomly selected positions and chromosomes, where \( p_m \) is the probability of mutation.

Step 6: Repeat steps 2–5 until a termination criteria is met. This can be the set maximum number of evolutions, the set minimum improvement of the best performance in successive generations, or a known global optimum. Moreover, Elitism, the inclusion of the best current set in the next population, is used throughout the paper.

The selection in step 3 leads to the replacement of the less useful with the more useful information in the population. The crossover in step 4 provides a method of searching for better combination of that information. The mutation in step 5 helps to maintain diversity and generate new useful genetic material. The above three operations are the important components of the GA which is considered effectively for the global optimization problem (1) and (2). However, the SGA’s implementing these three operations in every individual is time consuming, and the fixed mutation probability neglects the differences among various bits. Moreover, SGA with too large mutation probability is the higher location of \( R_2 \) in the crossover sequence. If the translocation operator is not performed, the transformation operator is then carried out. Subsequence \( R_2 \) is selected in the first half part of the crossover sequence, while \( R_4 \) with the same length as \( R_2 \) is located in the last half part. After implementing the crossover operator in the neural sequences, \( N/2 \) parents produce \( N \) offspring.

(3) The mutation operator based on RNA operations: Since the mutation is to maintain the diversity of the population and generate new genetic material, it is performed among the offspring produced by the neural ones and the deleterious ones, which are totally 3\( N/2 \) sequences. In Neuhauser and Krone (1997), there exist ‘hot spots’ and ‘cold spots’ in the DNA sequence model, i.e., the nucleotide bases in the ‘cold spots’ mutate more slowly than those in the ‘hot spots’, which accords with the fact that the spots in different bit positions have different effects on the solutions to the problem. Therefore, at the beginning stage of evolution, larger probability of mutation is assigned to those RNA nucleotide bases in the higher bit positions (‘the hot spots’), so the larger feasible region is explored. When the region of the global optimum is found, the mutation probabilities of RNA nucleotide bases in the higher bit positions are decreased to prevent better solutions from disrupting. The hot spots are correspondingly converted into the cold spots. The nucleotide bases of RNA sequence between 1 and \((L/2)\) are set as the low bit position, the left as the high bit position. Accordingly, there are two kinds of mutation probability \( p_{nh} \) and \( p_{nl} \), which are described as follows:

\[
p_{nh} = a_1 + \frac{b_1}{1 + \exp[aa(g - g_0)]} \tag{3}
\]

\[
p_{nl} = a_1 + \frac{b_1}{1 + \exp[-aa(g - g_0)]} \tag{4}
\]

where \( a_1 \) denotes the initial mutation probability of \( p_{nl} \), \( b_1 \) the range of transmutability, \( g \) the evolution generation, \( g_0 \) decides the generation where great change of mutation probability occurs and \( aa \) denotes the speed of change. The changing curves with evolution generation of \( p_{nh} \) and \( p_{nl} \) are shown in Fig. 1. The coefficients of (3)
and (4) are selected as follows: \( a_1 = 0.02, b_1 = 0.2, g_0 = G/2, \) and \( \text{aa} = 20/G. \)

After calculating the mutation probability in terms of (3) and (4), \( 4 \) decimal fractions between 0 and 1 are produced compared with the above probability. If the mutation probability is larger than the corresponding decimal fraction, the nucleotide base is replaced by one of another three integers, i.e., a random integer between 0 and 3 besides the nucleotide base itself. Thus, three mutations of nucleotide base in section 2.2 are implemented.

(4) **The selection operator:** There are now \( 3N/2 \) sequences, the new population is constructed by choosing the best \( N/2 \) sequences and the worst \( N/2 \) sequences. The proportional selection can be performed in the population with \( N \) sequences. The number of reproduction is calculated as follows.

\[
n_r = \left( \frac{J_i}{\sum_{i=1}^{N} J_i} \right) \times N
\]

where \( J_i = F_{\text{max}} - f_i, F_{\text{max}} \) is a constant chosen to guarantee \( J_i > 0 \). If \( n_r \) is equal to zero, one sequence is still reproduced. So, \( N \) sequences can be reproduced as the parents of the crossover operator.

Thus, the proposed RNA-GA utilizes some RNA operations and DNA sequence models under selection and mutation. From the above description, when the calculation process of electronic RNA-GA is replaced by the biological RNA molecular computing, even if good resolutions in RNA molecular computing are eliminated incidentally, better solutions can still be derived after recombination of the left solution next time. After finite repetitions, the optimal solutions or near-optimal solutions of RNA sequences increase greatly, and the right RNA sequence will be obtained relatively simply.

### 3. Global convergence analysis of RNA-GA

As for the global optimization problem (1) and (2), Li et al. (2002) made a summary of conditions guaranteeing the convergence of GA with mutation operator, which is listed as follows.

**Assumption 1.** At every generation \( t \), if every individual (\( x \)) in the population and a random individual \( y \) satisfy \( x \neq y \), then there exists \( p(t) > 0 \), where \( p(t) \) is the probability of changing \( x \) into \( y \) by one mutation operator.

**Theorem 1.** If GA with elitist strategy satisfies Assumption 1, it will converge in probability to the optimal solution of the problem. Moreover, its convergence is independent to the distribution of initial population.

Based on the assumptions and the theorem described above, the convergence of the proposed RNA-GA is analyzed as follows.

As defined in RNA-GA, RNA chains are essentially the quadruple sequences with length of \( L \). Its coding space is defined as \( S = \{0, 1, 2, 3\}^L \), i.e., \( |S| = 4^L \). Set \( N \) as the population of individuals, \( \Omega \) as the sequence set of RNA, \( X \) as the population composed of the elements in \( \Omega \), \( \Omega^N \) as all possible populations in \( \Omega \), \( \Omega^N = \{X_1, X_2, \ldots, X_n\} \).

The transition probability matrix \( (P) \) can be decomposed as the product of three probability matrices (Li et al., 2002): crossover \((C)\), mutation \((M)\) and selection \((S)\), i.e. \( P = CMS \).

Matrices \( C, M, S \) possess the following properties:

\[
\sum_{j=1}^{n} c_{ij} = 1
\]

where \( c_{ij} \) is the probability of changing sequence \( i \) into sequence \( j \) by the crossover operator.

\[
m_{jk} = \prod_{i=1}^{N} \left( \frac{P_m}{C-1} \right) H_i, k \in \{1, 2, \ldots, n\}
\]

where \( m_{jk} \) denotes the probability of turning sequence \( j \) into sequence \( k \) by mutation, \( C = 4 \), and \( H \) is the individual Hamming distance between sequence \( j \) and sequence \( k \).

Because at least one individual is to be selected, the following inequality holds:

\[
\sum_{k=1}^{n} s_{kj} > 0, \quad q \in \{1, 2, \ldots, n\}
\]

where \( s_{kj} \) denotes the probability of changing sequence \( k \) into sequence \( q \) by the selection operator. Let

\[
\rho = \left[ \min \left( \frac{P_m}{3}, 1 - P_m \right) \right]^{NL}
\]

Then, the following inequality is obtained based on (7) and (9).

\[
m_{jk} > \rho
\]

The following equation is derived in term of the running process of RNA-GA

\[
P = CMS = \sum_{k=1}^{n} \left( \sum_{j=1}^{n} c_{ij} m_{jk} \right) s_{kj}
\]
Table 1

Five test functions

<table>
<thead>
<tr>
<th>Test functions</th>
<th>Optimal solution</th>
<th>Optimal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2; \quad x_1, x_2 \in [-5.12, 5.12]$</td>
<td>(1, 1)</td>
<td>0</td>
</tr>
<tr>
<td>$\max f_2(x) = \left(\frac{a}{b + (x_1^2 + x_2^2)}\right)^2 + (x_1^2 + x_2^2); \quad a = 3.0, b = 0.05, x_1, x_2 \in [-5.12, 5.12]$</td>
<td>(0, 0)</td>
<td>3600</td>
</tr>
<tr>
<td>$f_3(x) = \sum_{i=1}^{2} -x_i \sin(\sqrt{</td>
<td>x_i</td>
<td>}); \quad x_1, x_2 \in [-500, 500]$</td>
</tr>
<tr>
<td>$f_4(x) = x_1 \sin(\sqrt{x_1^2 + 1 - x_2}) \times \cos(\sqrt{x_1^2 + x_2 + 1}) + (x_2 + 1) \cos(\sqrt{x_2 + 1 - x_1}) \times \sin(\sqrt{x_1^2 + x_2 + 1}); \quad x_1, x_2 \in [-512, 512]$</td>
<td>(-488.63, 512)</td>
<td>-511.7329</td>
</tr>
<tr>
<td>$f_5(x) = \left(\frac{(x_1 - 100)^2 + (x_2 - 100)^2}{4000}\right) - \cos(x_1 - 100) \cos\left(\frac{x_2 - 100}{\sqrt{2}}\right) + 1; \quad x_1, x_2 \in [-600, 600]$</td>
<td>(100, 100)</td>
<td>0</td>
</tr>
</tbody>
</table>

Substitute (10) into (11), the following inequality is obtained:

$$P \geq \sum_{k=1}^{n} \left(\sum_{j=1}^{n} c_{kj}\rho\right) \cdot s_{kj}$$  \hspace{1cm} (12)

Eq. (12) can be rewritten by substituting (6) into it:

$$P \geq \rho \sum_{k=1}^{n} s_{kj}$$  \hspace{1cm} (13)

From (8) with (13), the ultimate answer is:

$$P > 0$$  \hspace{1cm} (14)

Since (14) is satisfied, the finite states homogeneous Markov chain constructed by each generation is ergodic, which accords with the fact that the initial population of RNA-GA is finite. The transition probability of genetic operation independent of time $t$ is larger than zero. Hence, the operations of RNA-GA satisfy Assumption 1. Based on the above analysis, RNA-GA with elitist strategy can converge in probability to the optimal solution of the problem.

4. Performance of the RNA-GA

4.1. Test functions

In order to test and compare performances of the proposed optimization algorithms, a test environment must be provided in the form of several objective functions. Selecting a group of representative functions is not an easy task, since any particular combination of properties represented by a test function does not allow for generalized performance statements. Table 1 compiles a list of commonly used test functions, which represent a group of landscape classes with various characteristics: large search space, numerous local minima and fraudulence. All the functions are two-dimensional, which make it easy to visualize them and to see the algorithms in action.

The global optimum of Rosenbrock function $f_1$ locates in a very narrow valley with a flat bottom, which is difficult to follow. The non-separable characteristic of this quadratic function with differing eigenvalues further increases the hardness of the problem. Needle-in-haystack (NiH) function $f_2$ has 4 local optima with the value 2748.78, which is close to the global optimum (3600). The degree of fraudulence can be changed by selecting different coefficients of NiH function. Schwefel’s function $f_3$ is symmetric, separable and multimodal, its global minimum is near the bound of the domain and geometrically distant from the second minimum points. Therefore, the search algorithms are potentially prone to converge in the wrong direction. Rana function $f_4$ is a non-separable and highly multimodal function. Its best solutions are at the corners. Griewank function $f_5$ is highly multimodal with thousands of widespread local minima. However, the location of the minima is regularly distributed. It is hard to optimize these functions using classical methods, as well as most evolutionary algorithms. The successful search can only be expected by methods with effective anti-deceptive search ability.

4.2. Adaptability of the parameters

This subsection studies the adaptation ability of the main parameters in RNA-GA. It is applied to optimize Rana model $f_4$ as well as Griewank function $f_5$. The first group of experiments use fixed coefficients $aa = 20/G$ and $a_1 = 0.02$, but various values of $b_1$. The second use fixed $b_1 = 0.2$ and $a_1 = 0.02$, but various aa. Similarly, the third use fixed coefficients $aa = 20/G$ and $b_1 = 0.2$, but various $a_1$. As plotted in Figs. 2–4, the experiment results are given in the tendency curves of the average best-so-far objective function values over 50 independent runs.

The figures show that the linear convergence can be clearly observed. It reflects that the adaptation mechanism of RNA-GA operates effectively. It also illuminates that the convergence rate is more sensitive to $b_1$ and $a_1$ than $aa$, and larger $b_1$ and $a_1$ lead to faster convergence. However, for the sake of robustness, the more moderated settings of $a_1 = 0.001–0.05$, $b_1 = 0.1–0.3$, $aa = 20/G$ are recommended, since too large $b_1$ and $a_1$ are likely to result in random searching, while too small $b_1$ and $a_1$ are prone to converge prematurely. The feasibility of the recommended settings has also been empirically verified by the successful optimizations of the functions $f_1$ to $f_3$.

4.3. Comparisons between RNA-GA and SGA

To get an idea of how RNA-GA converge to the global optimum, the distributions of the individuals in the RNA-GA
populations on contour plots of the functions to be optimized are shown in the paper. When the proposed algorithm is used, the maximum generation number is limited to 1000, the initial population size $N$ is set to 60, and the individual length is set to 40. Other parameters are kept unchanged as shown in Section 3. For comparison, SGA in the GA toolbox of MATLAB 7.1 is adopted to optimize these functions. GA uses proportional selection, adaptation mutation and two-point crossover with the probability of 0.8. Elitist reservation mechanism is utilized to ensure the monotony of the best-so-far individuals in populations. The runs of the algorithms terminate when the predeterminated numbers of function evaluations are performed or the inequality $|F_b - F^*| < \Delta$ is satisfied, where $F_b$ denotes the objective function value of the best-so-far individual, $F^*$ the global optimum, and $\Delta$ is a precision requirement of the optimal solution, which is set to 0.0001. The behavior of RNA-GA and SGA are shown in Figs. 5–14. It is obvious that RNA-GA possesses the improved population diversity. With similar initial population, RNA-GA is capable of exploring more search space than SGA during the in-process population. Even at the end of evolution, the population of RNA-GA still remains in diversity, while the population of SGA is prone to converge to one point, which makes SGA tend to trap into local minima.

To obtain statistically significant data, a sufficiently large number ($R$) of independent runs must be performed. The performances of the convergence speed are measured by the average evaluation number $\bar{E}$, the minimum and maximum evaluation number $E_{\min}$ and $E_{\max}$ over $R$ runs, where $\bar{E} = (1/R)\sum_{i=1}^{R} E_i$, $E_i$ ($i \in 1, \ldots, R$) are the actual evaluation numbers satisfying
terminate conditions. The corresponding data obtained by the sample size $R = 50$ are listed in Table 2.

The global search ability is measured by $F_{\text{min}}, F_{\text{max}}$ and $\bar{F}$, which denotes the minimum, maximum and average optimal value of the test functions over $R$ runs, respectively. And the rate of the runs accurately reaching the global optimum is also demonstrated by Suc.rate (the abbreviation of success rate). The corresponding data are listed in Table 3.

The statistic results in Table 3 show that RNA-GA can conquer the fraudulence of test functions with fewer local optima (such as $f_2$ and $f_3$) and is successful in all tests. As for $f_5$, because there exist thousands of local optima with regularity, RNA-GA

<table>
<thead>
<tr>
<th>Test function</th>
<th>RNA-GA</th>
<th>SGA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$E$</td>
<td>$E_{\text{min}}$</td>
</tr>
<tr>
<td>$f_1$</td>
<td>614.45</td>
<td>109</td>
</tr>
<tr>
<td>$f_2$</td>
<td>489.5</td>
<td>327</td>
</tr>
<tr>
<td>$f_3$</td>
<td>323.5</td>
<td>117</td>
</tr>
<tr>
<td>$f_4$</td>
<td>760.1</td>
<td>493</td>
</tr>
<tr>
<td>$f_5$</td>
<td>497.0667</td>
<td>5</td>
</tr>
</tbody>
</table>
Table 3
Comparison of the global research ability by RNA-GA and SGA over 50 runs

<table>
<thead>
<tr>
<th>Test function</th>
<th>RNA-GA</th>
<th>SGA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$F$</td>
<td>$F_b$</td>
</tr>
<tr>
<td>$f_1$</td>
<td>8.570e−7</td>
<td>8.336e−9</td>
</tr>
<tr>
<td>$f_2$</td>
<td>3.600e+3</td>
<td>3.600e+3</td>
</tr>
<tr>
<td>$f_3$</td>
<td>−837.966</td>
<td>−837.966</td>
</tr>
<tr>
<td>$f_4$</td>
<td>−511.613</td>
<td>−511.733</td>
</tr>
<tr>
<td>$f_5$</td>
<td>0.0020</td>
<td>4.156e−8</td>
</tr>
</tbody>
</table>

Fig. 7. Behavior of RNA-GA in the NiH problem.

Fig. 8. Behavior of SGA in the NiH problem.
also traps into local optima. However, most of trials can find the global optimum. As for Rana function \( f_4 \), RNA-GA find the global optimum located at \((-488.63, 512)\) with value of \(-511.7329\), which is different from the optimum in references \((512, 512)\) with value of \(-511.7011\). However, the success rate is still quite small because of thousands of local optima. Compared with the statistic results of SGA, the global search ability of RNA-GA is greatly improved. As for \( f_1 \) and \( f_3 \), SGA has similar success rate as RNA-GA in Table 3, however, the convergence speed of SGA is quicker than RNA-GA in Table 2, since there exists a turning point of mutation probability at evolution number \( g_0 \), the rapidity of convergence is sacrificed to obtain the diversity of population. As for the single model function \( f_1 \), though both SGA and RNA-GA can find the optimal results in the 100% success rate, the results of RNA-GA are superior to those of SGA. Hence, RNA-GA has better search performances than SGA.

5. Simulations on parameter estimation

Due to the superior performance of RNA-GA, such a hybrid strategy is applied for model parameter estimation in this section. The following model

\[
y(t) = g(u(t), \theta) \tag{15}
\]
is considered, where \(y(t)\) is the system output, \(u(t)\) the system input vector, and \(\mathbf{\theta} = [\theta_1, \theta_2, \ldots, \theta_k]^T\) are the parameters to be estimated, and the form of model \(g\) is supposed to be known. The job is to estimate parameters \(\mathbf{\theta} = [\theta_1, \theta_2, \ldots, \theta_k]^T\) according to certain index that is a function of the true system outputs and the model sample outputs under certain system inputs. Obviously, such a problem can be cast as numerical optimization problem, where the design variables are \(\mathbf{\theta} = [\theta_1, \theta_2, \ldots, \theta_k]^T\), and the objective function can be defined as follows:

\[
f(\mathbf{\theta}) = \sum_{t=0}^{n_s-1} |y(t) - \hat{y}(t)|
\]

where \(n_s\) is the sample length. Model parameter estimation can be illustrated by Fig. 15, where the algorithm is used to adjust parameters to be estimated based on objective value. Because of the non-linearity and other complicated characteristics of chemical processes, traditional methods of parameter estimation are not applicable. Here, the RNA-GA is employed for such a role. The larger value of \(n_s\) is, the more precise the parameter estimation of RNA-GA, while the smaller of \(n_s\) is, the more sensitive RNA-GA to time delay. Hence, \(n_s\) is selected between 20 and 100.

5.1. Parameter estimation for a heavy oil thermal cracking three lumping model

The identification plant—a heavy oil thermal cracking three lumping model is given by Song et al. (2003), which is described...
as follows.

\[
x_L = \frac{K_{LP0} e^{-E_{LP}/T}}{n_L} [1 - (1 - z)^{n_L}]
+ \frac{K_{WP0} K_{WLP0} e^{-(E_{WP} + E_{WLP})/T}}{n_W - K_{WLP0} e^{-E_{WLP}/T}}
\times \left\{ \frac{1 - (1 - z) K_{WLP0} e^{-E_{WLP}/T}}{K_{WLP0} e^{-E_{WLP}/T}} + \frac{1}{n_W} [(1 - z)^{n_W} - 1] \right\}
\]

(17)

where \( z \) and \( T \) are the input variables and \( x_L \) is the system output. RNA-GA is used to estimate parameters of the above model: \( K_{LP0}, K_{WP0}, K_{WLP0}, E_{LP}, E_{WP}, E_{WLP}, n_L, n_W \). Twenty groups of data are chosen randomly from Song et al. (2003) to estimate the unknown parameters. Here, the optimization index function is selected as:

\[
f = \sum_{i=0}^{n_s-1} |x_L(i) - \hat{x}_L(i)|
\]

where \( \hat{x}_L \) is obtained by using (17).

The bounds of parameters to be estimated are set as follows: \( K_{LP0} \in (0, 10), K_{WP0} \in (0, 10), K_{WLP0} \in (0, 10), E_{LP} \in (800, 1500), E_{WP} \in (1500, 4000), E_{WLP} \in (1500, 4500), n_L \in (0, 5), n_W \in (0, 5) \). The maximum evolution number of RNA-GA is set as 2000, and the other parameters of RNA-GA are the same as shown in test function optimization. Run the RNA-GA independently 50 times for the above model, the best results are listed in Table 4, where the comparisons between RNA-GA based results and SGA-based results (Song et al., 2003) are also provided. The
Table 4
Results of parameter estimation by SGA and RNA-GA

<table>
<thead>
<tr>
<th>Methods</th>
<th>$K_{LP0}$</th>
<th>$K_{WP0}$</th>
<th>$K_{WLP0}$</th>
<th>$E_{LP}$</th>
<th>$E_{WP}$</th>
<th>$E_{WLP}$</th>
<th>$n_L$</th>
<th>$n_W$</th>
<th>$F_b$</th>
<th>$f_{(test\ data)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGA (Song et al., 2003)</td>
<td>4.680</td>
<td>5.155</td>
<td>4.197</td>
<td>1257</td>
<td>1850</td>
<td>3776</td>
<td>1.191</td>
<td>1.488</td>
<td>0.4883</td>
<td>1.2116</td>
</tr>
<tr>
<td>RNA-GA</td>
<td>3.221</td>
<td>9.333</td>
<td>3.385</td>
<td>998.1</td>
<td>2779</td>
<td>3662</td>
<td>1.439</td>
<td>3.308</td>
<td>0.2815</td>
<td>0.8738</td>
</tr>
</tbody>
</table>

best fitness value ($F_b$) of SGA is calculated by using the same data as RNA-GA’s. In addition, the corresponding model fitting curves obtained by SGA and RNA-GA using the training test are shown in Fig. 16.

All of 56 groups of data provided by Song et al. (2003) are used as test samples to verify the efficiency of above model parameters. The model fitting curves are shown in Fig. 17, where the value of index function using SGA is 1.2116, while it is 0.8759 using RNA-GA. From Figs. 16 and 17 and their corresponding values of index function, the fitting precision of RNA-GA is superior to that of SGA.

5.2. Parameter estimation of FCC unit main fractionator by RNA-GA

5.2.1. Process description of FCC unit main fractionator

Fluid catalytic cracking (FCC) is an important oil refinery process, which converts high molecular weight oils into lighter hydrocarbon products. It consists of reactor–regenerator, riser reactor, main fractionator, absorber–stripper–stabilizer, main air blower, wet gas compressor, etc. As one of the most important parts of FCC unit, main fractionator is critical to realize the advanced control of FCC unit. The main fractionator configuration of a 1.4 million tonnes FCC unit in an oil refinery factory is shown in Fig. 18.

The oil quality is conventionally controlled by the top temperature and the temperature of tray 20. However, the temperature cannot precisely reflect the product quality, which mainly depends on the parting accuracy in the boiling range, such as the end point of crude oil and the pour point of light diesel oil. Therefore, the quality index of main fractionator in this work is not the temperature but the end point of naphtha. An on-line soft sensing for end point of naphtha has been established successfully by the members in our laboratory (Zhong & Wang, 1998).

The 400 °C oil gas from the reactor is fed into the bottom of the main fractionator at tray 1 after heat removal. Once in contact with 275 °C counter flow of slurry from top circulation, the oil gas is cooled down, and separated into gas, crude oil, light diesel oil, cycle oil and slurry. To provide enough inner reflux and make load distribution uniform, the fractionator has 4 heat circulation systems, i.e., top heat removal circulation, the first mid heat removal circulation, the second mid heat removal circulation and slurry heat removal circulation. In the slurry heat removal system, the slurry is extracted from tower bottom and exchanges heat quantity with fuel oil. It is then separated into two parts: one as cycle slurry, the other is discharged out the fractionator. In the second heat removal circulation system, there exist three parts: the first part return tray 2 as an inner circulation; the second part return tray 5; the third part is extracted as cycle oil. The first heat removal circulation system locates between tray 17 and tray 20, the light diesel oil is drawn out from tray 20. In the top heat removal circulation system, oil gas is abstracted from tray 29, and return tray 32 when the temperature cools down to 80 °C.
When the oil gas enters the tower top, the vapor phase and liquid phase (crude oil) can then be obtained.

5.2.2. Process modeling

From the above analysis, the influencing factors of end point are top temperature, top pressure, top heat removal, etc. The top heat removal is the main method to adjust the end point. The factors which affect the pour point are mainly top load changes, top pressure changes, first and second mid heat removal. Since most of heat in the second mid circulation acts as thermal resource of bottom reboiler, only a small amount of heat is used to adjust the temperature of tray 1, the first mid heat removal becomes the most important adjustment method for pour point. The change of heat removal is implemented by the change of flow or by the change of temperature. Therefore, the top circulation flow, the first mid flow and the second mid flow are chosen as the manipulated variables, denoted as MV1–MV3; the top temperature, end point of crude oil and pour point of light diesel oil are selected as controlled variables, denoted as CV1–CV3.

To implement the advanced control of product quality for main fractionator, the dynamic model is necessary to be established. To be applicable in industrial spot, the model structure cannot be too complicated, which is supposed as the following discrete representation: \((a_1 + a_2 z^{-1})/(1 - b z^{-1})z^{-d_1}\). Because the coupling mainly exists between CV1 and CV2, as well as CV2 and CV3, the FCC unit main fractionator MIMO process models can be simplified as shown in Table 5.

The MIMO process model is then be obtained as follows:

\[
CV_1(z^{-1}) = \frac{a_{111}(1) + a_{112}z^{-1}}{1 + b_{11}z^{-1}} z^{-d_{11}} MV_1(z^{-1}) + \frac{a_{211}(1) + a_{212}z^{-1}}{1 + b_{21}z^{-1}} z^{-d_{21}} MV_2(z^{-1})
\]  

(18)
Table 5
FCC unit main fractionator process models

<table>
<thead>
<tr>
<th></th>
<th>CV1</th>
<th>CV2</th>
<th>CV3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MV1</td>
<td>( \frac{a_{11}(1) + a_{11}(2)z^{-1}}{1 - b_{11}z^{-1}} ) (-d_{11}) 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MV2</td>
<td>( \frac{a_{22}(1) + a_{22}(2)z^{-1}}{1 - b_{22}z^{-1}} ) (-d_{22}) ( \frac{a_{23}(1) + a_{23}(2)z^{-1}}{1 - b_{23}z^{-1}} ) (-d_{23})</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MV3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6
Results of estimated parameters for main fractionator by RNA-GA

<table>
<thead>
<tr>
<th></th>
<th>Numerator</th>
<th>Denominator</th>
<th>Delay</th>
<th>( F_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CV1</td>
<td>( a_{11} = [0.0904, -0.0045] )</td>
<td>( b_{11} = -0.9138 )</td>
<td>( d_{11} = 6.0000 )</td>
<td>1.1913</td>
</tr>
<tr>
<td>CV2</td>
<td>( a_{21} = [0.0626, -0.0789] )</td>
<td>( b_{21} = -0.0012 )</td>
<td>( d_{21} = 3.0000 )</td>
<td>0.8636</td>
</tr>
<tr>
<td>CV3</td>
<td>( a_{22} = [0.0626, -0.0196] )</td>
<td>( b_{22} = -0.4165 )</td>
<td>( d_{22} = 8.0000 )</td>
<td>0.5439</td>
</tr>
<tr>
<td>CV4</td>
<td>( a_{23} = [0.4284, -0.3602] )</td>
<td>( b_{23} = -0.9254 )</td>
<td>( d_{23} = 1.0000 )</td>
<td></td>
</tr>
<tr>
<td>CV5</td>
<td>( a_{32} = [0.5171, -0.3801] )</td>
<td>( b_{32} = -0.8599 )</td>
<td>( d_{32} = 3.0000 )</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 19. Comparison of model prediction and real value for CV1.

Since there exists the coupling of the estimated parameters in (18) and (19), it is difficult to estimate these parameters. RNA-GA in this work can be used to solve the complicated parameter estimation problem. In Zhong, Zhang, and Wang (2001), the model of FCC unit main fractionator was established according to typical field data by the members of our laboratory. Therefore, the input–output data are produced by the model provided in Zhong et al. (2001). Moreover, based on the knowledge of refin-
5.2.3. Parameter estimation for modeling by RNA-GA

The objective functions for the above three equations are listed as follows.

\[ f_i = \sum_{k=1}^{L} |CV_i(k) - \hat{CV}_i(k)|, \quad i = 1, 2, 3 \]  

(21)

where \( \hat{CV}_i \) is the model output. The input data are a group of step signals, and the corresponding outputs are generated by the given model with a maximum divergence of \( \pm 10\% \). The input and output data are normalized between 0 and 1, and the parameter domain of the fraction equations is set as \([-1, 1]\). Since it is a stable process, the range of denominator coefficient is reduced as \([-1, 0)\), and the range of time delay is supposed as \([1, 10]\).

All parameters of RNA-GA are the same as in parameter estimation for the heavy oil thermal cracking three lumping model. Fifty independent runs of RNA-GA for each \( f_i \) are implemented. The results with the best value of objective functions are selected as the ultimate estimated parameters, which are listed in Table 6.

The model outputs (CV1, CV2 and CV3), as well as those of the real process for parameter estimation, are shown in Figs. 19–21. To verify the efficiency of the obtained model, another group of test data is selected, which is shown in Figs. 22–24. The figures illuminate that the parameter estimation using RNA-GA is applicable in real processes, and the established model can factually describe the dynamic characteristic between manipulated variables and controlled variables. So it can be concluded that RNA-GA is an effective and efficient approach for model parameter estimation of this kind of chemical processes.

6. Conclusions

By combining RNA operations and DNA sequence model with genetic algorithm, a framework of RNA-GA is proposed for complex function optimization as well as model parameter estimation. Numerical simulation results demonstrate the effectiveness of the hybridization, especially the advantages of RNA-GA in terms of optimization quality, efficiency as well as initial conditions. The superiority of the proposed RNA-GA is the combination of DNA sequence model with variable mutation probability as well as the combination of multiple various RNA operators. Theoretically, the proposed algorithm can be applied to real biochemical reaction after simple transition, and the brute force method of DNA computing can be broken, as all operators in RNA-GA are obtained by RNA molecular operations. However, at present its feasibility of laboratory experiment still lies on biochemical techniques of DNA computing.

Acknowledgements

This paper has been supported by the National Natural Science Foundation of China under grants 60421002 and 70471052. The authors would also like to thank associate professor J.M. Zhang, L. Xie for discussions about this work, and the anonymous reviewers for their helpful comments.
References
