Virus-enhanced Genetic Algorithms Inspired by DNA Computing

Jan J. Mulawka, Piotr Wąsiewicz, Katarzyna Piętak

Warsaw University of Technology, Nowowiejska 15/19, Warsaw, Poland
{jml,pwas}@ipe.pw.edu.pl

Abstract. DNA computing is a new promising paradigm to develop an alternative generation of computers. Such approach is based on biochemical reactions using DNA strands which should be carefully designed. To this purpose a special DNA sequences design tool is required. The primary objective of this contribution is to present a virus-enhanced genetic algorithms for global optimization to create a set of DNA strands. The main feature of the algorithms are mechanisms included specially for searching solution space of problems with complex bounds. Formulae, describing bounds of power of sequences' sets, which satisfy criteria and estimation functions are expressed. A computer program, called Mismatch, was implemented in C++ and runs on Windows NT platform.

1 Introduction

1.1 The development of molecular computing

Traditional computer systems are implemented with electronic networks comprising transistors. This technique has been under development for recent 50 years. A couple of years ago however, a new technique of computing based on molecules (called molecular computing) has appeared [1,2,3]. In this approach different chemical compounds can be used to store and process information. The calculations are performed in vitro by adequate interpretation of chemical reactions. Investigations indicate that organic substances used in biology are well suited for this purpose, especially nucleic acids and proteins. For example DNA molecules provide high degree of selectivity of biochemical reaction and therefore can serve as an information carriers. The approach based on DNA molecules is an interesting alternative for electronic computation.

The molecular method of computing has been initiated by Adleman [1] who demonstrated how to solve NP-complete combinatorial and graph problems which are difficult for conventional computers. A number of possible applications of DNA computing has been repored so far. For example it has been demonstrated that DNA computing is suitable for programming in logic [4]. Expert systems can also be realized by this technique. In [5] implementation of the inference engine based on a backward chaining

algorithm has been discussed. However, other paradigms of reasoning also may be used.

A single-strand has a phospho-sugar backbone and four bases Adenine, Thymine, Cytosine, Guanine denoted by the symbols A, T, C, and G, respectively. A double-strand may be formed of two single-strands due to hybridization or in other words annealing reaction, because A is complementary with T and C is complementary with G. Due to this reaction the oligonucleotides may connect with each other forming longer double-strand DNA chains.

1.2 Potentials of DNA computing

Molecular technology offers the following potential advantages over the classical electronic computers.

Speed of operation. In a single test tube there may exist 10^{20} DNA strands. Average reaction time of biological operation performed on a strand is 1000s. The speed of parallel operations [6] performed may amount $V=10^{14}$ MIPS. By automatization of the biological operations and by reducing the reaction time below 1000s gives a chance to speed up the performance. It should be noted that existing supercomputers comprising a couple of thousand of processors may achieve the speed up to 10^6MIPS (ASCI Red - 1,8Tflops, 9256 processors) [7].

Data capacity. Assuming that in molecular computing particular strand is coded with about 500-800 bits, data memory realized on DNA strands in a single test tube ($1dm^3$ of water solution comprises approximately 10^{20} strands) may amount to 1 bit/nm^3. It allows to achieve the memory about 10^8 times greater than current magnetic memories ($250*10^9$ bit/$inch^2$). A concept of DNA memory exceeding about 10^6 times human memory has been proposed by Baum [8]. An important feature of such approach is that the DNA memory is of an associative type.

Energy saving. In molecular technology the calculations are performed on DNA molecules. For biochemical reactions adequate temperatures are required. The energetic efficiency of these operations is high. It amounts approximately 10^{19} op/J [9] comparing to 10^9 op/J for traditional supercomputers.

2 Limitations of DNA computing

DNA computing is based on the technology of genetic engineering which suffers a number of drawbacks. DNA operations initiated by a human are using biochemical reactions prone to errors [10,11]. Unexpected results in these reactions are due to many factors such as not exact temperature, specifity of the chemical reaction performed in a solution, etc. The main factors which cause mistakes of biological operations are the following:

Extraction - 5 % errors. It may happen that during extraction unproper strand is selected. To decrease the chance for such mistake it is proposed to keep a constant amount of reacting strands in a vessel. It may be accomplished by replicating the strands during reaction. Some improvements are also achieved by making change in coding manner, by taking double quantity of the input data, etc.

Replication by PCR (Polymerase Chain Reaction) - 0,001 % errors. During biochemical reactions strands may be replicated with an error caused by different speeds in connecting complementary nucleotides.

Ligation. Sometimes it happens that ligase enzyme causes connections of unproper strands. To improve this mistake it is proposed to select the proper enzyme for a given set of DNA strands.

Annealing. In ideal case during annealing two single-strands are connected to produce double-strand of DNA. However, occasionally single strands are folding and anneal with themself, or particular pairs of strands create irregular shapes. These are so called hybridization mistakes. To eliminate annealing errors the proper coding of input data is recommended.

As follows from the above considerations in molecular computing the proper encoding and selection of DNA strands is necessary to minimize the disadvantages of biological technology.

3 Description of virus-enhanced genetic algorithm for design of DNA strands

Genetic algorithms [12] belong to techniques that can be successfully applied to NP-hard optimization problems e.g. for optimizing functions with many local and one global optima. They consist of selections and different types of genetic operations which are repeated in cycles. It is assumed that genetic algorithms search for the global optimum [13] of such functions with complex constraints. In addition our algorithm implements an idea of viruses (autonomous DNA strands) to optimization problems. We assume two different types of viruses: fitness and reproduction ones. Each of the former has a unique fixed transition position in an overwritten DNA strand and a fixed length. The latter are created during reproduction and they have no fixed position in a victim strand and random lengths. Both types have a fixed lifetime, but the latter can die if they are not effective in action this means in improving the fitness of individual by overwriting the part of its original chromosome (in order to be copied with the infected individual and infect other).

The algorithm of Fig. 1 has been applied in the task of choosing DNA sequences for molecular computing. In this problem the sequence of nucleotides is binary coded. Each of these nucleotides is described by two bits as shown in Fig. 2. They are ordered by their atomic weights.

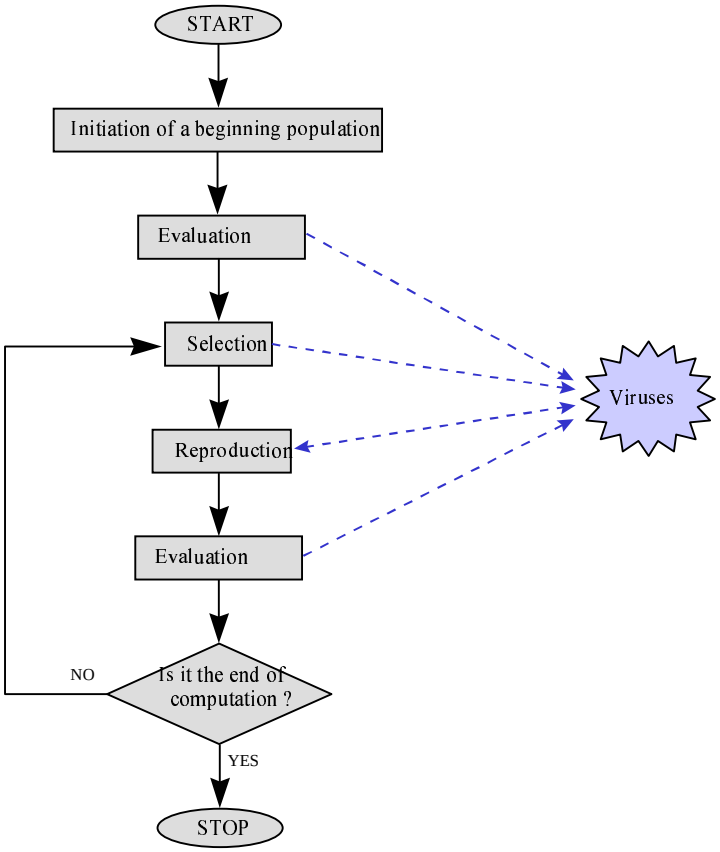

Fig. 1. The structure of a virus enhanced genetic algorithm

```
C -------> 00 -------> 288,2 [dalton]
T -------> 01 -------> 303,2 [dalton]
A -------> 10 -------> 312,2 [dalton]
G -------> 11 -------> 328,2 [dalton]
```

Fig. 2 Binary coding of nucleotides

Thus every four nucleotides are coded in one byte. In Fig. 3 a process of their coding is explained. The advantages of this approach are the following: maximum density of information makes computation more easy, one change of a bit causes a change of a non-complementary type of a nucleotide, very quick testing of complementary strings with a logic function XOR as shown below.

C ≡ G complementary pairs 00 XOR 11 = 11
T = A complementary pairs 01 XOR 10 = 11

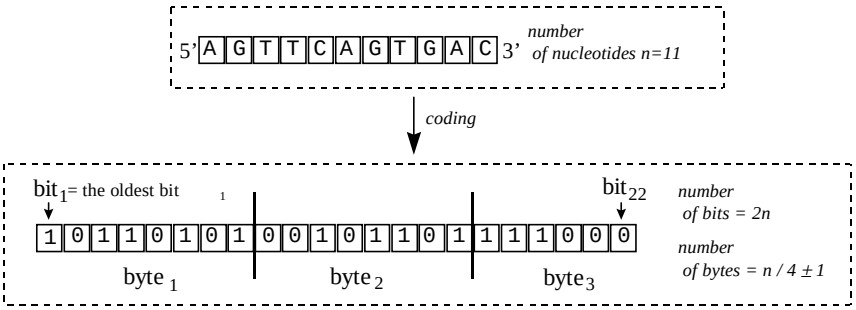

Fig. 3 The process of coding DNA strands into binary strings

Here a fitness of such strand is a number of strand mishybridizations with itself and other strands. The less mishybridizations the better value of this fitness function is obtained. It can be easy described when it is only the largest mishybridization taken into consideration and such the fitness function of a single oligonucleotide e.g. with number one is as follows

$$f_1(a_i) = -\left(100 \cdot k_{\max} + 50 \cdot not\_unique - c_{\min}\right)$$

where $f_1(a_i)$ - a fitness function of a DNA strand, k_{max} - length of the longest mishybridization, c_{min} - the smallest value of a fitness function in a strand population, ***not_unique*** - equals 1 if there are more mishybridizations with the same length, otherwise - 0. It is important to have always values above 0, therefore c_{min} is subtracted.

5 Results of Experiments

In our experiments the first task was to find one DNA strand of η nucleotides with the smallest own complementarity. The solution has to be energeticly stabilized. This means the capacity of GC pairs (%GC) has to be equal to 50% with ± *GCMaxInequality[%]*. During computation these strands are transformed into 2η binary strings and are decoded only when they are presented on a screen or written to a file. In order to have solutions within constraints connected with energy (%GC) and weight (%CT) stability before analyzing of such a string fitness this string has to be repaired.

$$\begin{cases} |gc(a_i)| \leq GCMaxInequality \\ |ct(a_i)| \leq CTMaxInequality \end{cases}$$

Table 1. Results of optimal *100-nucleotide* strand computation.

Genetic Operations	No of trials	Without virus operations		With virus operations	
		a fitness function value	a number of fitness function evaluations	a fitness function value	a number of fitness function evaluations
Crossover mutation	1	-400	9700	-350	16600
	2	-400	800	-400	11800
	3	-400	11600	-350	32600
	4	-400	2300	-400	7300
	5	-350	17200	-350	8400
uniform crossover mutation	1	-350	45700	-350	17700
	2	-350	49300	-400	3800
	3	-400	11500	-350	4900
	4	-400	13300	-400	400
	5	-350	23000	-350	9000
uniform crossover subinversion	1	-350	1567	-350	2848
	2	-350	3829	-350	4485
	3	-450	100	-350	3158
	4	-350	6611	-350	6120
	5	-350	2025	-350	2920

The experiments were performed with the following values of parameters of analysis

MinMatch	3	and ones of the genetic algorithm:	
EnergeticRestrict	1		
GCMaxInequality	10	GA_PopulationSize	100
CTMaxInequality	15	GA_NumberOfGenerations	500
		MutationProbability	0.1
		CrossoverProbability	0.4
		RVirusCreationProbability	0.5
		VirusInfectionProbability	0.2

and for $\eta=100$ and five trials, the results are depicted in Table 1.

In Fig. 5 it is shown how to process the analysis of selfcomplementarity of a strand.

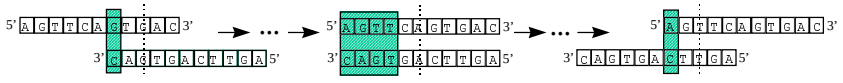

Fig. 5 Strand selfcomplementarity analysis

CAGACACGGCCACTCACCGGACTCTCACAGAAGCGCTCACGGAAAGATGGAA
TACTCGGGACAGAACCACAGCAAACAGGAACACACTACATACAGACGT

Fig. 6 One of the best strands selected

One of the best strings found during optimization is depicted in Fig. 6. Value of the function fitness is equal to 350 (the longest mishybridization of three nucleotides and there are several the same sequences).

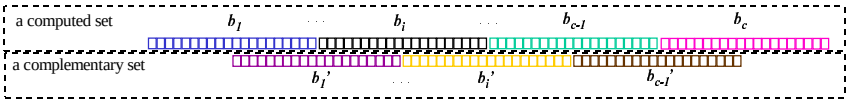

Fig. 7 The only allowed DNA set hybridization

The next task of the optimization was to find a set of $|c|$ DNA η-*nucleotide* strands which hybridize only in one way (Fig. 7) . The process of coding into one $2|c|\eta$ binary string is depicted in Fig. 8. Its fitness function is described by the formula shown below.

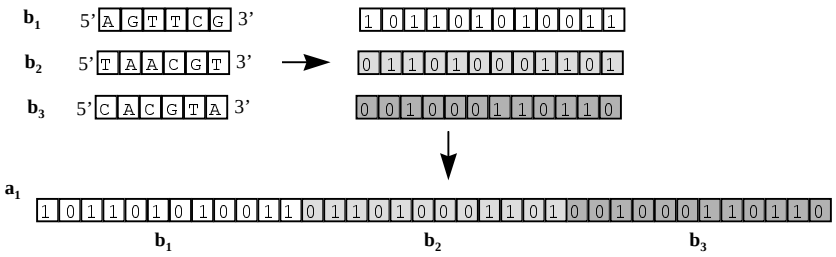

Fig. 8 Binary coding of a DNA strand set into one binary string

$$f_2(a_i) = -\left(100 \cdot k_{all\,\max} + 20 \cdot others\_count\right) + f_{1\,\max}(b) + \frac{\sum_{j=1}^{c} f_1(b_j)}{c}$$

where $f_2(a_i)$ - a fitness function of a DNA strand, $\boldsymbol{k_{allmax}}$ - length of the longest mishybridization in the strand set, ***others_count*** - equals 1 if there are more mishybridizations with the same length in the strand set, otherwise - 0 (for > 4 equals 4), $\boldsymbol{f_{1\,max}(b)}$ - the largest value of the single strand fitness function, $\frac{\sum_{j=1}^{c} f_1(b_j)}{c}$ - average of the single strand fitness function in the strand set.

With the same bounds and parameters for virus operations:

MaxVirusLife	10,	parameters for GA:	
WeekVirus	5,		
RVirusCreationProbability	0,	GoodComplementarity	2,
(these viruses cause mishybridizations)		GA_PopulationSize	20

and for $\eta = 20$ the results described in Table 2 were obtained.

Table 1. Results of optimal 20-*nucleotide 20*-strand *set* computation.

Genetic Operations	f2: a value of DNA set fitness	max fl max. value of the single strand fitness	avg fl avg. value of the single strand fitness	max length of mishybridi-zation	a number of fitness function evaluations
uniform	-1140.0	-350	-110.0	6	6300
crossover	-1175.0	-350	-125.0	7	4400
mutation	-1190.0	-400	-130.0	6	1200
without reproduction viruses					
uniform	-1132.5	-350	-102.5	6	6000
crossover	-1165.0	-350	-95.0	7	10050
mutation	-1152.5	-350	-142.5	6	10900
without reproduction viruses					

The output of the program - a set of optimized DNA strands:

ACAACCTTGACGCAATTCGT	TACTAATAGAGGAAGCATCG
GTGTTGAAGGGTCTTATGTA	TAAAGCTCGGTCCCCCTGTA
GAGAAGAGGCCGTGTTCGAG	TTCGGCGAAGAATGTGCCCA
CGAAAGGCTTGGCAGGATTA	TTATCGAGTACAAGACCTAG
CATGAAGAAGTCGAACGGTA	GGTGGAGCTACGAGTCTGTT
TTGAGCTGCACTGCTTGAAC	AGTGAGAGCCGCCTATCTTA
GGCTGCTCCTGAACGTAGTT	GCAACTTACTAGCGCTATAT
TTGAGACGGCGGTACTGAGA	AGCCTACCAATGCAATCAAT
GAATCCTCCTCTAGTAGGCC	TTACGCTCTATCACACGAAC
CGCTGATGTACTGTGTTGTC	AATGATGTCTCAACGTGTCT

6 Conclusions

We have presented the method of optimizing DNA strands suitable for DNA computing by special designed genetic algorithms. In description of molecular

computing, applications, profits and problems relevant to DNA-based technology were discussed and need for creation of a special DNA sequences design tool is explained. Changes in sequences of oligonucleotides can be well-controlled by viruses. They are to remember single strands with small own complementarity and within constraints connected with weight and energy stability. Viruses can have a variable length or are equal to 2η and overwrite oligonucleotides. Proper design of oligonucleotide sequences is very important for further development and applications of DNA computing.

Acknowledgments

This work was supported by the KBN Grant No 8T11F00816.

References

[1] L.M. Adleman, Molecular Computation of Solutions to Combinatorial Problems, Science 266 (11/1994), 1021-1024
[2] J.J. Mulawka, Molecular Computing - promise for new generation of computers, Proc. Workshop on Circuit Theory, Signal Processing and Appl., Budapest, 94-99, Warsaw University of Technology, 1997
[3] J.J. Mulawka, P. Wąsiewicz, Obliczenia molekularne - nowy kierunek technik informacyjnych, Informatyka 7/8, 1998
[4] M. Ogihara, A. Ray, Simulating Boolean Circuits on a DNA Computer, University of Rochester, USA, 1996
[5] J.J. Mulawka, P. Borsuk , P. Węgleński: Implementation of the inference engine based on molecular computing technique, Proc. IEEE Int. Conf. Evolutionary Computation (ICEC'98), Anchorage 493-498(1998)
[6] L.M. Adleman, On Constructing a Molecular Computer, University of Southern California, Los Angeles, USA, 1995
[7] T. Glanville, The Intel ASCI Red Supercomputer, 1997
http://www.frontiernet.net/~tglanvil/ASCI/Report.htm
[8] E. Baum, Building an Associative Memory Vastly Larger Than the Brain, Science, vol.268, 583-585, April 1996
[9] T.A. Bass, Gene Genie, 1995
http://www.cs.princeton.edu/~dabo/biocomp/molecular.html
[10] K. Langohr, Sources of Error in DNA Computation, University of Ontario, 1997
[11] R. Lipton et al., Making DNA Computers Error Resistant, Princeton University, 1996
[12] D.E. Goldberg, Algorytmy genetyczne i ich zastosowania, WNT, 1995
[13] Z. Michalewicz, Algorytmy genetyczne + struktury danych = programy ewolucyjne, WNT, 1996