DNA SIMULATION OF GENETIC ALGORITHMS: FITNESS COMPUTATION[1]

Maria Calvino, Nuria Gomez, Luis F. Mingo

Abstract: In this paper a computational mode is presented base on DNA molecules. This model incorporates the theoretical simulation of the principal operations in genetic algorithms. It defines the way of coding of individuals, crossing and the introduction of the individuals so created into the population. It resolves satisfactorily the problems of fitness coding. It shows also the model projection for the resolution of TSP. This is the basic step that will allow the resolution of larger examples of search problems beyond the scope of exact exponentially sized DNA algorithms like the proposed by Adleman [Adleman, 1994] and Lipton [Lipton, 1995].

Keywords: Genetic Algorithms, Fitness Function, DNA Computing, Evolutionary Computing.

ACM Classification Keywords: I.6. Simulation and Modelling, F.m. Theory of Computation

Introduction

Deoxyribonucleic acid (DNA) is the chemical inside the nucleus of all cells that carries the genetic instructions for making living organisms. A DNA molecule consists of two strands that wrap around each other to resemble a twisted ladder. The sides are made of sugar and phosphate molecules. The "rungs" are made of nitrogen-containing chemicals called bases. Each strand is composed of one sugar molecule, one phosphate molecule, and a base. Four different bases are present in DNA - adenine (A), thymine (T), cytosine (C), and guanine (G). The particular order of the bases arranged along the sugar - phosphate backbone is called the DNA sequence; the sequence specifies the exact genetic instructions required to create a particular organism with its own unique traits. Each strand of the DNA molecule is held together at its base by a weak bond. The four bases pair in a set manner: Adenine (A) pairs with thymine (T), while cytosine (C) pairs with guanine (G). These pairs of bases are known as Base Pairs (bp).

DNA and RNA computing is a new computational paradigm that harnesses biological molecules to solve computational problems. Research in this area began with an experiment by Leonard Adleman [Adleman, 1994] in 1994 using the tools of molecular biology to solve a hard computational problem. Adleman's experiment solved a simple instance of the Travelling Salesman Problem (TSP) by manipulating DNA. This marked the first solution of a mathematical problem with the tools of biology.

Computing with DNA generated a tremendous amount of excitement by offering a brand new paradigm for performing and viewing computations. The main idea is the encoding of data in DNA strands and the use of tools from molecular biology to execute computational operations. Besides the novelty of this approach, molecular computing has the potential to outperform electronic computers. For example, DNA computers may use a billion times less energy than electronic computers, while storing data in a trillion times less space. Moreover, computing with DNA is highly parallel: in principle there could be billions upon trillions of DNA or RNA molecules undergoing chemical reactions, that is, performing computations, simultaneously. Some advantages of DNA are that it is both self-complementary, allowing single strands to seek and find their own opposite sequences, and can easily be copied. Also, molecular biologists have already established a toolbox of DNA manipulations, including restriction enzyme cutting, ligation, sequencing, amplification, and fluorescent labelling, giving DNA a head start in the arena of non-silicon computing.

Despite the complexity of this technology, the idea behind DNA computing springs from a simple analogy between the following two processes, one biological and one mathematical: the complex structure of a living organism ultimately derives from applying sets of simple instructed operations (such as copying, marking, joining,

[1] This work has been partially supported by Spanish Project **TIC2003-9319-c03-03** "Neural Networks and Networks of Evolutionary Processors".

inserting, deleting, etc.) to information in a DNA sequence, and computation is the result of combining very simple basic arithmetic and logical operations.

The fact that the definition of gene implies the concept of a unit of minimum relative information as far as a functional unit and that it corresponds to the structural unit of basic molecular DNA and by association can be considered as the basic unit of mutation and of heredity, has taken it directly to trying to simulate genetic algorithms using DNA.

Overview of Genetic Algorithms

Genetic Algorithms are adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic. The basic concept of GA is designed to simulate processes in natural system necessary for evolution, specifically those that follow the principles first laid down by Charles Darwin of survival of the fittest. As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem.

First pioneered by John Holland in the 60s [Holland, 1975], Genetic Algorithms has been widely studied, experimented and applied in many fields in engineering worlds. Not only does GAs provide an alternative method to solving problem, it consistently outperforms other traditional methods in most of the problems link. Many of the real world problems involved finding optimal parameters, which might prove difficult for traditional methods but ideal for GAs.

GAs are based on an analogy with the genetic structure and behaviour of chromosomes within a population of individuals using the following foundations:

- Individuals in a population compete for resources and mates.
- Those individuals most successful in each 'competition' will produce more offspring than those individuals that perform poorly.
- Genes from ‘good' individuals propagate throughout the population so that two good parents will sometimes produce offspring that are better than either parent.

Thus each successive generation will become more suited to their environment. After an initial population is randomly generated, the algorithm evolves the through three operators: selection which equates to survival of the fittest; crossover which represents mating between individuals; mutation which introduces random modifications.

Selection Operator: gives preference to better individuals, allowing them to pass on their genes to the next generation. The goodness of each individual depends on its fitness. Fitness may be determined by an objective function or by a subjective judgement.

Crossover Operator: prime distinguished factor of GA from other optimization techniques. Two individuals are chosen from the population using the selection operator. A crossover site along the bit strings is randomly chosen. The values of the two strings are exchanged up to this point. If S1=000000 and s2=111111 and the crossover point is 2 then S1'=110000 and s2'=001111. The two new offspring created from this mating are put into the next generation of the population. By recombining portions of good individuals, this process is likely to create even better individuals.

Mutation Operator: with some low probability, a portion of the new individuals will have some of their bits flipped. Its purpose is to maintain diversity within the population and inhibit premature convergence. Mutation alone induces a random walk through the search space. Mutation and selection (without crossover) create parallel, noise-tolerant, hill-climbing algorithms.

One of the basic arguments of the theory of evolution is that individuals that show similarities are related. Based on this principle, Holland observed that certain groups of individuals with particular similarities in some positions in their chains had good common properties whilst others were worse. Abstracting this idea Holland defines the concept of scheme (H) in one binarian coding with chains of length ℓ, thus;

$$H = h_{\ell-1} \dots h_0 \in \{0, 1, *\}^{\ell} \leftrightarrow H = \{ s_{\ell-1} \dots s_0 / h_j \neq * \rightarrow s_j = h_j \}$$

In other words, one scheme represents a certain subgroup of the population in which the individuals differentiate themselves at most in the position of the asterisks.

For example, the scheme H = 10 ∗ 01∗ correspond with the chain group
{100010, 100011, 101010, 101011} At the same time any group of chains defines a scheme, suffise to consider the J-th Coordinate.

$$\pi_j : \{0, 1\}^{\ell} \to \{0, 1\}$$
$$s_{\ell-1} \ldots s_0 \mid\to s_j$$

And to define the scheme H thus;

$$h_j = \begin{cases} 0 \text{ if } \pi_j(H) = \{0\} \\ 1 \text{ if } \pi_j(H) = \{1\} \\ * \text{ if } \pi_j(H) = \{0, 1\} \end{cases}$$

In fact, the group of chains that can be generated by crossing the elements of the group C is exactly H. For example if C = {001011, 011111} then each chain of H = 0 * 1 * 11 can be generated by crossing the elements of C, even 011011 y 001111, that were not initially in C.

Obviously in some schemes their elements show more likeness between themselves than in others. To quantify this idea there are two concepts; The order of a scheme which is the number of fixed alleles in the scheme and the length of definition which is the distance between the first and the last of the fixed alleles. For example if H = 00 * *1*, then o(H) = 3 y δ(H) = 4.

In essence, Holland's scheme theorem affirms that the algorithm drives the search for the optimum through certain subgroups of the population. In other words, explores the space of search through those areas that on average are more adequate.

Given that during the reproduction a chromosome is selected with a probability proportional to fitness

$$\frac{f(s)}{\sum_{s \in P(t)} f(s),}$$

where P(t) denotes the population t-th. Then if we start from a population of N elements the spected number of representatives of H in the following iteration is

$$E(n(H, t+1)) = n(H, t).\ N.\ \frac{f(H, t)}{\sum_{s \in P(t)} f(s),}$$

where **E** denotes the operator hope, n(H, t) is the number of chains in the scheme
In the generation t-th and

$$f(H, t) = \frac{\sum_{s \in H} f(s)}{n(H, t)}$$

is the average value of the scheme in that generation. If we now consider the action of the operators of crossing and mutation the previous equation is transformed in:

$$E(n(H, t+1)) = n(H, t).\ \frac{f(H, t)}{\overline{f}}\ [1 - \alpha(H)]$$

where:

$$\overline{f} = \frac{\sum_{s \epsilon H} f(s)}{N}$$

Represents the average fitness of the population and **α(H)** depends of the structure of H and the probabilities of crossing and mutation, **pc** y **pm,** but not of **t**, Because:

$$\alpha(H) = p_c \cdot \frac{\delta(H)}{\ell - 1} \cdot (1 - p_m)^{o(H)}$$

This expression, ignoring the terms of grade ≥ 2 in Newton's binomial, and because pm _ 0.01, turns into the final formula of the fundamental Theorem of genetic algorithms (or Theorem of schemes):

$$E(n(H, t+1)) = n(H, t) \cdot \frac{f(H, t)}{\overline{f}} [1 - p_c \cdot \frac{\delta(H)}{\ell - 1} - o(H) \cdot p_m]$$

Thus, we have that if H is a scheme with a fitness level greater tan the average of the population it is hoped to increase the number of chains with the structure of H in the following generation as long as a(alpha) is small. In other words, the principle of this theorem's can be interpreted saying that "short schemes of lower order with greater fitness than the average increase the number of representatives in the successive generations" This type of schemes that seem to play an important role in the way that GAs act, are known as building blocks.

It seems then tha by juxtaposing solid blocks of small size, increasingly better individuals could be built. This leads us to think that functions that can be defined utilising short schemes of lower order and high suitabilty would be easy to optimise for the Gas [Mitchell, 1994]

This affirmation, known as the hypothesis of the building blocks [Holland, 2000] seems very reasonable. In fact, GAs have been designed for various applications are empirical evidence that for different types of problems such hypothesis is correct.

DNA Simulation of Genetic Algorithms

The construction of a genetic algorithm for the resolution of an optimisation problem requires the definition of the genetic architecture. In this sense the election the manner of coding of the individuals represents an important point to obtain the correct solution.

The said coding must be done in a way that each chain allows the storage of information corresponding to an individual of a genetic algorithm. [González, 2002] Later on the bits will be represented independently of the position that they are in. [Lipton, 1995]

Given that the genetic composition of an individual constitutes a multyfactorial variable, the definition of the individual within a population will be dependent of the problem to be dealt with, that is, an individual has a genome that must comply with a number of prerequisites subordinate to the problem, to be considered suitable within that population.

The process of representation of the genes as a minimum unit of information requires the analysing of the problem in question with the purpose of stipulating the number of bits assigned to the mentioned gene.

A gene will be represented as a whole of three fields; the percentage of Timine of the gene will be proportional to the fitness that represents the central field:

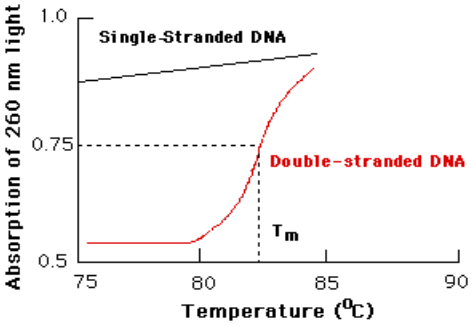
ENC(X) FITNESS xy ENC(Y)

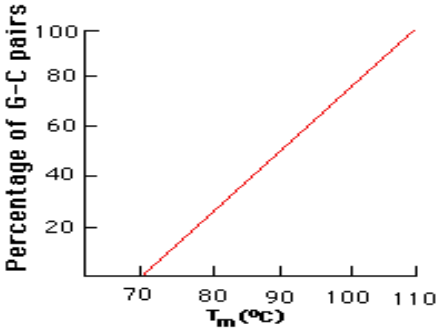
With the purpose of mapping such fields for later processing the recombining DNA will bring a linker between fields in a way that can be recognised by restriction endonucleases. [Bonen, 1996]

Given that PCR will be used for the amplification of the DNA, each individual will carry at the same time, at both ends of the chain a sequence that will hybridise with a specific primer in the annealing phase (hybridization).

The fitness must be embedded in the coding of the individuals and given its definition will be determined by the

content in G+C which implies that the fitness of an individual will be directly related with the fusion temperature and hence would be identifiable by spectophotometry (A_{260}) and separable by electrophoresis techniques (DGDE) [Macek 1997].





It is possible then to detect the perfect candidate by means of DGDE as it would be the one the possible candidates to present the greater number of GC pairs and therefore has the greater electrophotometric mobility.

The identification of the individuals of the population requires the tailing of the recombinant with a specific field that identifies the individual in a unique manner, the coding of this field will be done by means of the following function:

$$\text{CODE: } N \rightarrow \{G, C\}^*$$

Where N is the totallity of the natural numbers. It will receive a number corresponding to an individual within a population and returns a sequence of nucleotides.

The identification of the individuals in the mating zone requires again the inclusion of a field (Nm) which will be determined again by the function CODE.

The generation of the initial population has as an objective obtaining an aleatory population with a number of individulas equal to the size of the population. The complexity of the sinthesis of the sequence will be directly related to the number of genes used for the representation of the individuals within the genetic algorithm.

Basically it consists of a recombinant by means of a union of compatible fragments digested with restriction endonucleases.

The final format would look as follows:

PCR-primer Np REp XY RE0 XY RE1 … REn-1 XY REp Np -1 PCR-primer

The selection of individuals will be done by means of specific probes of the problem in question and the isolation of the individuals will be achieved by means of electrophoretic techniques (DGDE).

After selection, the individual will be introduced in the mating zone. For this he must be modified adding a specific field of such zone. In the event that a crossing of individuals is required, it is done in a temporal test tube containing the pair of individuals.

The mutation will be induced on each of the individuals results of the crossings operation in the genes in which the mutation frequency surpasses others obtained at ramdom and consists in the substitution of a gene by its complement in bases.

Later ill adapted individuals from the mating zone will be eliminated and will be substituted by the created recombinants. To determine the finalisation of the algorithm in each iteration the average of population adaptation is calculated. Once the convergence of the population is reached the best individual will be analysed by means of radioactive marking (o etching) techniques.

Fitness Computation on TSP Problem

The TSP is interesting not only from a theoretical point of view, many practical applications can be modelled as a travelling salesman problem or as variants of it, for example, pen movement of a plotter, drilling of printed circuit boards (PCB), real-world routing of school buses, airlines, delivery trucks and postal carriers. Researchers have tracked TSPs to study biomolecular pathways, to route a computer networks' parallel processing, to advance cryptography, to determine the order of thousands of exposures needed in X-ray crystallography and to determine routes searching for forest fires (which is a multiple-salesman problem partitioned into single TSPs). Therefore, there is a tremendous need for algorithms.

In the last two decades an enormous progress has been made with respect to solving travelling salesman problems to optimality which, of course, is the ultimate goal of every researcher. One of landmarks in the search for optimal solutions is a 3038-city problem. This progress is only party due to the increasing hardware power of computers. Above all, it was made possible by the development of mathematical theory and of efficient algorithms.

There are strong relations between the constraints of the problem, the representation adopted and the genetic operators that can be used with it. The goal of travelling Salesman Problem is to devise a travel plan (a tour) which minimises the total distance travelled. TSP is NP-hard (NP stands for non-deterministic polynomial time) - it is generally believed cannot be solved (exactly) in time polynomial.

TSP Solution using a DNA Genetic Algorithms Simulation. Fitness Computation.

Applying the previous protocol to the TSP of four cities in which the total size of the population is 256 (N) and the number of arches 6 (M) the individuals will be coded with an amount T inversily proportional to the length of the arches. The resulting sequence is shown in fig 1:

arch	Distance	Number of nucleotides	Fitness	Nucleotides of G
AB	1	40	1	40
BC	1	40	1	40
CD	4	40	4	10
BD	3	40	3	13
AD	2	40	2	20
AC	6	40	6	6

Figure 1.- Results of TSP simulation

The final format would look like this:

PCR-primer Np REp XY RE0 XY RE1 … REn-1 XY REp Np-1 PCR-primer

where XY ϵ { AB, BC, CD, AC, BD }

The selection criteria of the individuals in this case involves the encoding for an existing way to do this the strands of selection join two vertex of the graph neither initial nor final.

To be discarded are those individuals that do not start in the original city, those that do not finish in the final city as well as those that are found in repeated cities.

Selected are those structures which in the stretch O to N are covered by the strands of selection.

The format for the introduction of individuals in the mating zone is the following:

PCR-primer Np Nm REp XY RE0 XY RE1 … REn-1 XY REp Np-1 PCR-primer

where XY ϵ { AB, BC, CD, AC, BD }

We proceed then to the crossing mutation and evaluation of the degree of adaptation of the individuals prior to their introduction to the population and the process is repeated until the convergence of the population obtaining in each iteration the degree of adaptation

Conclusion

The generation of this work has produced a new approach to the simulation of genetic algorithms with DNA. The problem of fitness evaluation in a parallel form has been resolved satisfactorally. This does not imply that the definition would be independent of the problem at hand even though there are rules that facilitate such definitions and that can be solved by means of genetic algorithms.

In a GA simulated with DNA the concept fitness field disappears.

The coding of the individuals is closely related with the characteristics of electrophoretic migration.

The fitness of the individual is embedded in his coding and any attempt to add such field represents a grave error. The addition of such field would mean a personalisation of the individual thus preventing a massive and anonymous parallelism.

Bibliography

[Adleman, 1994] Leonard M. Adleman. Molecular Computation of Solutions to Combinatorial Problems. Science (journal) 266 (11): 1021–1024. 1994.

[Adleman, 1998] Leonard M. Adleman. Computing with DNA. Scientific American 279: 54-61.1998.

[Amos, 2005] Martyn Amos.Theoretical and Experimental DNA Computation, Springer. ISBN 3-540-65773-8. 2005.

[Baum, 1996] Eric B. Baum, Dan Bohec. Running dynamic programming algorithms on a DNA computer. Proceedings of the second Annual Meeting on DNA Based Computers. 1996.

[Bonen, 1996] Dan Boneh, Christopher Dunworth, Richard J. Lipton, and Jiri Sgall. On the Computational Power of DNA. DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science 71. 1996.

[Darwin C] Charles Darwin. On the Origin of the Species by the Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life. Murray, London, 1859.

[González, 2002] Eduardo González Jiménez, Valery I. Poltev. La simulación computacional de procesos genéticos a nivel molecular. Ciencia y Cultura. Elementos, 479(47): September- November. 2002.

[Holland, 1975] J.H.Holland. Adaptation in Natural and Artificial Systems. MIT Press. 1975.

[Holland, 2000] Holland, J. H. (2000). Building blocks, cohort genetic algorithms, and hyperplane-defined functions. Evolutionary Computation, 8:4, 373-391.

[Kari, 2000] Lila Kari, Greg Gloor, Sheng Yu. Using DNA to solve the Bounded Post Correspondence Problem. Theoretical Computer Science 231 (2): 192–203. 2000.

[Lipton, 1995] Richard J.Lipton. Using DNA to solve NP-Complete Problems. Science, 268:542-545. April 1995. [Paun, 1998] Gheorge Paun, Grzegorz Rozenberg, Arto Salomaa. DNA Computing - New Computing Paradigms, Springer-Verlag. ISBN 3540641963. 1998.

[Mitchell, 1994] Mitchell, M., Holland, J. H., & Forrest, S. (1994). When will a genetic algorithm outperform hill climbing? IAdvances in Neural Information Processing Systems 6, 51-58, MIT Press.

[Macek 1997] Milan Macek M.D. Denaturing gradient gel electrophoresis (DGDE) protocol. Hum Mutation 9: 136 1997.

Authors' Information

Maria Calvino – Dep. Inteligencia Artificial. Facultad de Informática, Universidad Politécnica de Madrid, Boadilla del Monte, 28660 Madrid, Spain; e-mail: maria.calvino@medicimage.com

Nuria Gomez – Dep. Organización y Estructura de la Información, Escuela Universitaria de Informática, Universidad Politécnica de Madrid, 28031 Madrid, Spain; e-mail: ngomez@dalum.eui.upm.es

Luis Fernando de Mingo Lopez – Dep. Organización y Estructura de la Información, Escuela Universitaria de Informática, Universidad Politécnica de Madrid, 28031 Madrid, Spain; e-mail: lfmingo@eui.upm.es