Clustering over DNA Strings*

NURIA GOMEZ BLAS
Escuela de Informática
Dept. OEI - UPM
Crta. de Valencia km. 7
28031 Madrid - Spain

EUGENIO SANTOS
Escuela de Informática
Dept. OEI - UPM
Crta. de Valencia km. 7
28031 Madrid - Spain

MIGUEL ANGEL DIAZ
Escuela de Informática
Dept. OEI - UPM
Crta. de Valencia km. 7
28031 Madrid - Spain

Abstract: This paper presents an unsupervised learning algorithm over strings that can be applied in aminoacid sequences. This way a set of aminoacids is obtained and they represent the minimal distance to the whole *DNA* pool. Biological laboratory protocols over this cluster could be easily studied instead over the whole pool and protocols could be used in the whole pool. This process can help to develop protocols applicable in the *DNA* computing paradigm. A brief review of *DNA* computing is also shown.

Key–Words: DNA Computing, String Clustering, Unsupervised Learning, Protocols

1 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, see figure 1.

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 [1] in 1994 using the tools of molecular biology to solve a hard computational problem. Adlemans experiment solved a simple instance of the *Traveling Salesman Problem* (TSP) by manipulating *DNA*. This marked the first solution of a mathematical problem with the tools of biology.

1.1 Enzymes

In molecular biology, enzymes play an important role as catalysts of reactions, and transformers of information encoded in the *DNA* molecules. Several enzymes have played an important role in *DNA* computing, and as tools for the manipulation and transformation of *DNA* molecules, will continue to do so

*This work has been partially supported by Spanish Grant TIC2003-09319-c03-03.

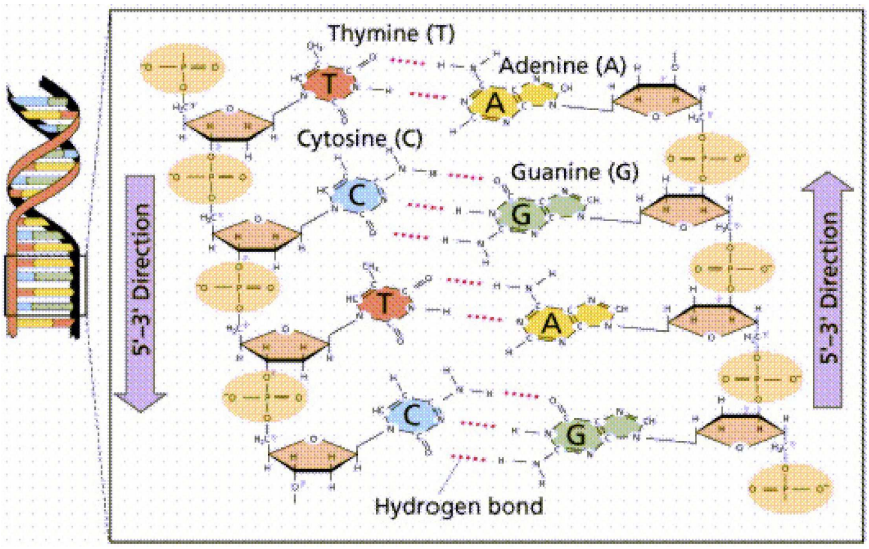

Figure 1: *DNA* Molecule

[2].

Restriction enzymes arise in natural systems as part of the cell's defense mechanisms, in which they cut up *DNA* that is not protected by specific cheicals, or methylated. Typically, a restriction enzyme recognizes a specific nucleotide sequence in a double-stranded *DNA* point. Another enzyme is ligase, which aids the formation of the phosphodiester linkages in double-stranded *DNA*. Other useful enzymes are exonucleases and endonucleases which chop *DNA* into its component nucleotides from the end of strand, or location internal to a strand, respectively.

An important technique in molecular biology is the *Polymerase Chain Reaction - PCR*. The enzyme polymerase synthesizes a *DNA* strand in the $5'$ to $3'$ direction on a *DNA* template. *PCR* is used for amplification of specific base sequences. In the basic technique, double-stranded *DNA* is separated, and short oligonulceotide primers are hybridized to the target. Of course, this necessitates that part of the desired sequence be known for determination of the primers. Then, new *DNA* is synthesized from the primers using a polymerase. at this point, the amount of *DNA* has doubled. Then, additional primers are added to delineate the other end of the target sequence, and the entide process is repeated. Because of its repetivive nature, *PCR* has been automated, and is capable of generating a huge number of copies of the target sequence.

Gel electrophoresis[3] is a technique used to separate and visualize *DNA* molecules based on their size. It explotis the fact that the *DNA* molecules are electrically charged. The gel is a matrix of molecules with holes of varying size. Under an applied electric field , *DNA* molecules will move a distance that depends on their size, with smaller molecules moving farther. By comparison to standards, the sizes of molucules under analysis are determined. With appropiate cutting of molecules and standards, *DNA* molecules can be sequenced with this technique.

2 DNA Computing

Solving a problem with *DNA* must be done in a laboratory in order to exploit the whole advantages of parallelism and recombination. *DNA* computing process can be summarize in these steps:

Generate *DNA* strands corresponding to the coded individuals of the initial population Some information must be coded into *DNA* in order to solve a problem therefore, a combination of nucleotides in a *DNA* strand are needed. An specific codification must be defined for a given problem according to biological operations that is, sticky ends, the way enzymes work, and biological operations. For example, next string represents a *DNA* strand that code an individual, it has two sticky ends:

```
.....CGTAACCGTACCCAGACGTACGTAACCGT
TGCATGCATTGGCATGGGTCTGCATGCAT.....
```

Apply enzymes in order to start the recombination process The way *DNA* strands join or are modified depends on the application of enzymes: manipulation and transformation of *DNA* molecules. Most usual enzymes are: restriction enzymes, polymerase, ligase, exonucleases and endonucleases (see section 1.1). After applying an enzyme, previous string could result in two new *DNA* strands:

```
.....CGTAACCGTACCC
TGCATGCATTGGCAT...
-----------------
...AGACGTACGTAACCGT
GGGTCTGCATGCAT.....
```

Extract the solution Usually, gel electrophoresis is used to separate *DNA* molecules, based on their size, knowing the number of nucleotides of desired solution.

Next section gives an example of the *Travelling Salesman Problem* using *DNA* computation.

2.1 TSP DNA Algorithm

The following sequence of steps, which are adapted from Adleman's *DNA* algorithm, will extract the shortest route beween two cities S and C.

1. Assign a unique *DNA* sequence for each city.
2. Construct *DNA* representations of binary paths between two cities X and Y as follows, where n is the distance between the two cities.
3. To form longer routes out of binary paths, splice two binary paths P_1 and P_2 together if and only if the final city code for P_1 matches the first city node for P_2. Delete half of the code of the final city node in P_1 and half of the code of the first city in P_2.
4. To prevent loops, do not form longer routes if the final city in P_2 matches the first city in P_1.
5. Repeat the above two steps until no more routes can be formed.
6. Place all the *DNA* sequences produced so far into a test tube.
7. Extract all those routes which start with the *DNA* code of the desired start city and place in a separate test tube.
8. Extract all those routes which end with the *DNA* code of the desired destination city and plase in a separate test tube.
9. Sort all the remaining routes by lengh. Gel electrophoresis can be used here. The shortest *DNA* sequence represents the shortest route between the desired start and destination cities. Also, the sequence of *DNA* codes in the shortest strand provides information as to the order in which cities are encountered, and the length of the route can be calculated.

String $\in s$	$\Delta(u_i, c)$	Cluster c
		ABCDEFGH
BHHBHGFGBEBFEGABFGCDGFGAECHFEHFDHFFGDA		ABCDEFGH
FEADGEFFAAAHDFGHEEFFGC	B	ACDEFGH
CEAHBFEFEAADBDFGFGGHAB	B	ACDEFGHB
ECEBBHBEBDEEABDAGGCH	F	ACDEGHB
EEGFBHCFFEBFHHFHAFHE	FD	ACEGHB
FCEECGDDFGDDAEHHCCFADACHF	FDB	ACEGH
HHCHFBAEEDHAFEEFHBGAFCBDHA	FBD	ACEGHD
DHBABHAECDHHCFABBHDFCBACABFDGFDHGE	BF	ACEGHDF
BDBDBAEGEFHADGHFBAHGEAGBH	BC	AEGHDF
CDAECGGBEEBCBHEHGGEGCDBEDAFEH	CB	AEGHDFB
CCHBGBCCFCFAFABGHGGCBBBHHED	C	AEGHDFBC
HGHBCCADEDBFEGBBDBFBEEF		AEGHDFBC
GAAHHCDABGHDDFAAEBGBDG		AEGHDFBC
BFBDEGEGEFCACAEGBHBGEEEHHHABADEGCFAFG		AEGHDFBC
ADACBFHDCDBAGFHHFHAFEGACCGDCA		AEGHDFBC
EBHFCDADBFGACBFBFGFHHCBDFACBDF		AEGHDFBC
DFDAGAFEDHBEECEEGHHE		AEGHDFBC
HFEADDCBBABHBFFBGCGBCBEDF		AEGHDFBC
CCGAGCGGHFGHCACDFGCGD	EB	AGHDFBC
DEEDCHBCHACFGBGDFBHFDGBD	E	AGHDFBCE

Table 1: One iteration in the learning algorithm with 20 strings (8 symbols/aminoacids).

3 DNA Clustering

The whole process of *DNA* computing must be performed in a laboratory, that is, using test tubes, applying enzymes, heating or cooling test tubes in order to produce some reactions among the strands. A priori knowledge of these conditions might help biologist to adequate protocols to perform such tasks, computation with *DNA*.

The following learning algorithm can be apply over strings representing the aminoacid sequence in a *DNA* pool (that is grouping three by three nucleotides) in order to obtain one or more clusters (aminoacids set) that represents the whole *DNA* pool.

Given two strings u and v, a distance $\delta(u, v)$ is defined as a set containing strings that appear in u and do not appear in v. A more general distance measure can be defined as $\Delta(u, v) = \delta(u, v) \cup \delta(v, u)$, in this case some mathematical properties are accomplished, let $u = abcde$, $v = age$ and $w = hiaj$ then:

- $\Delta(u, v) = \{b, c, d, g\} = \Delta(v, u) = \{g, b, c, d\}$
- $\Delta(u, \Delta(v, w)) = \{a, g, h, i, j, b, c, d\} = \Delta(\Delta(u, v), w) = \{g, b, c, d, h, i, j, a\}$
- $\Delta(u, w) = \{b, c, d, e, h, i, j\} = \Delta(\Delta(u, v), \Delta(v, w)) = \{b, h, i, j, e, g, c, d\}$
- $\Delta(u, u) = \{\lambda\}$

An unsupervised learning over a set of strings $s = \{u_1, u_2, \cdots, u_n\}$ can be approximating taking into account unsupervised learning, defined in Self Organizing Maps, in order to obtain a string c such

as:

$$\sum_{i=1}^{n} |\Delta(u_i, c)| \leq \sum_{i=1}^{n} |\Delta(u_i, k)| \quad \forall k \neq c \qquad (1)$$

Next, the learning algorithm is proposed as follows:

1. Take any arbitrary cluster c (for example $c = u_p$ or $c = \lambda$).
2. For each string $u_i \in s$:
   (a) Let $p = \Delta(u_i, c)$.
   (b) If $|p| \leq |c|$ then an arbitrary symbol belonging to p must be removed from c provided $|\delta(u_i, c)| > 0$, if $|\delta(u_i, c)| = 0$ then a symbol belonging to $\delta(u_i, c)$ must be added to c.
   (c) If $|p| \not\leq |c|$ then an arbitrary symbol belonging to p must be added to c provided $|\delta(c, u_i)| > 0$, if $|\delta(c, u_i)| = 0$ then a symbol belonging to $\delta(c, u_i)$ must be removed from c.
3. This last process must be repeated over set s until a desired cluster length is reached.

A cluster is obtained when the algorithm stops. This cluster represents the aminoacid set that has the minimum distance with the whole *DNA* pool. Biological operations over such set might be apply over the whole pool.

4 Conclusions

A clustering algorithm over symbolic information has been proposed. An aminoacid set is obtained when applying over *DNA* strands. This set could be studied in order to find some biological protocols that can be used in the whole *DNA* set. Such protocols would be of special interest in the field of *DNA* computing. This algorithm is focused as a tool to help biologists.

More clustering algorithms could be implemented in order to take into account the aminoacid sequence in a *DNA* strand and the cardinality of a given aminoacid in a sequence.

References:

[1] Leonard M. Adleman.*Molecular Computation Of Solutions To Combinatorial Problems*. Science (journal) 266 (11): 10211024. (2004).

[2] Russell Deaton, Max Garzon et al. *DNA computing: A review*. Fundamenta Informaticae 30, 23-41. IOS Press. (1997).

[3] Old, R. W. and Primose, S. B. *Principles of Gene Manipulation*. Blackwell Scientific. (1985).

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

[5] 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).

[6] Gheorge Paun, Grzegorz Rozenberg, Arto Salomaa. *DNA Computing - New Computing Paradigms*, Springer-Verlag. ISBN 3540641963. (1998).

[7] Lila Kari, Greg Gloor, Sheng Yu. *Using DNA to solve the Bounded Post Correspondence Problem*. Theoretical Computer Science 231 (2): 192203. (2000).