DNA SYNTHETIC SEQUENCES GENERATION USING MULTIPLE COMPETING MARKOV MODELS

Diogo Pratas, Carlos A. C. Bastos, Armando J. Pinho, António J. R. Neves, Luís M. O. Matos

Signal Processing Lab, IEETA / DETI
University of Aveiro, 3810–193 Aveiro, Portugal
{pratas,cbastos,ap,an,luismatos}@ua.pt

ABSTRACT

The development and implementation of computational models to represent DNA sequences is a great challenge. Markov models, usually known as finite-context models, have been used for a long time in DNA compression. In a previous work, we have shown that finite-context modelling can also be used for sequence generation. Furthermore, it is known that DNA is better represented by multiple finite-context models. However, the previous generator only allowed a single finite-context model to be used for generating a certain sequence. In this paper, we present results regarding a synthetic DNA generator based on multiple competing finite-context models.

***Index Terms*—** Statistical generator, synthetic sequences, Markov models, DNA entropy.

1. INTRODUCTION

DNA generation is the process of sequence design, bringing into existence synthetic DNA sequence data, obtained by indirect measurement, based on pseudo-random models. In this way, synthetic sequence generation has now an important role in better understanding some biological characteristics. On the other hand, looking for DNA generation algorithms is also a way of finding models that describe the information source associated to DNA.

Some methods have been proposed to date. As reviewed by Shin et al. [1], those strategies rely on exhaustive search [2], random search [3], template-map [4, 5], graph methods [6], stochastic methods [7], dynamic programming [8], biologically-inspired methods [9, 10] and evolutionary algorithms [11, 12, 13].

In a previous work [14] we have proposed a method for DNA generation, relying exclusively on the Markov property. It uses a model that captures the statistical information along the sequence in order to generate the next symbol. As we know from previous work [15], DNA sequence data is better represented by multiple finite-context models, since DNA sequence data are non-stationary. For this reason, we have improved the previous DNA generator using multiple competing finite-context models.

This paper is organized as follows. In Section 2, we describe our algorithm. In Section 3, we provide experimental results, including an information content analysis and the Kullback-Leibler divergence assessment between the training sequences and the corresponding synthetic sequences. Finally, in Section 4, we draw some conclusions.

2. THE GENERATION METHOD

DNA sequence data are non-stationary. In fact, one of the reasons why most DNA encoding algorithms use a mixture of two methods, one based on repetitions and the other relying on low-order finite-context models, is to try to cope with the non-stationary nature of the data. We also follow this line of reasoning, i.e., that of using different models along the sequence. However, unlike the other approaches, we use exclusively the finite-context paradigm for modelling the data, changing only the order of the model as the characteristics of the data change. More precisely, we explore an approach based on multiple finite-context models of different orders that compete for generating the data.

For convenience, the DNA sequence is partitioned into non-overlapping blocks of fixed size, which are then generated by one of the finite-context models (the block generating model). Using several models with different orders allows a better handling of DNA regions with diverse characteristics. Therefore, although these multiple models are continuously updated, only the generating model is used to generate a given region (block). This model is chosen according to another finite-context model, that operates on an alphabet with size equal to the number of models, and that collects information regarding the sequence of models that are used along the DNA sequence. We call this sequence of models the side information.

The generation proceeds in two phases. In the first phase, a training DNA sequence is used for loading the statistics into the several finite-context models. During this phase, the counters that collect the statistics are continuously updated. The idea is to obtain a set of finite-context models that, because they have been exposed to the statistical properties of the training sequence, afterwords are able to generate synthetic sequences with similar properties. In the second phase, the models are kept frozen. This is the generating phase, where the synthetic sequence is created according to the statistics previously learned. Moreover, whereas the training phase is deterministic, because it consists on collecting the regularities of the training sequence, the generating phase is stochastic.

Figure 1 shows an example where two competing finite-context models are used. In this example, each model collects statistical information from a context of depth $k_1 = 5$ and $k_2 = 11$, respectively. At time n, the two conditioning contexts are $c_1 = x_{n-k_1+1} \cdots x_{n-1}x_n$ and $c_2 = x_{n-k_2+1} \cdots x_{n-1}x_n$. For additional details concerning the finite-context models see, for example, [16, 17, 15].

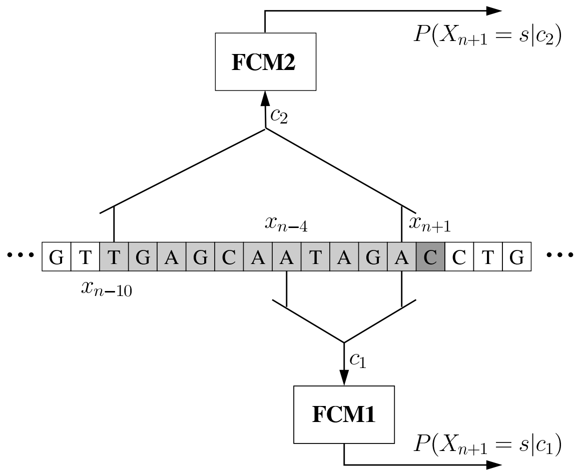

Fig. 1. Example of an overall model for estimating probabilities using multiple finite-context models, in this case two. The probability of the next outcome, X_{n+1}, is conditioned by the k_1 or k_2 last outcomes, depending on the finite-context model chosen for handling that particular DNA block. In this example, $k_1 = 5$ and $k_2 = 11$.

3. EXPERIMENTAL RESULTS

In this study, we used all the chromosomes from the human genome. However, due to space restrictions, we only present results regarding four chromosomes (1, 7, 21 and Y). The genome was obtained from the National Center for Biotechnology Information (NCBI) (`ftp://ftp.ncbi.nlm.nih.gov/genomes/H_sapiens/April_14_2003`), *Homo sapiens*, Build 33.

In a first phase, we generated three sequences (A, B, C), based on the first million of bases of human chromosome 1. In a second phase, we generated four sequences (D, E, F, G), based on the twenty second million of bases of chromosome Y. The main reason for using these sequences is their high repetitiveness [18] and consequently better coverage of low entropy regions. These are the regions that are more difficult to represent by the finite-context models and, therefore, we wanted to see how the generator would behave in this case.

Table 1. Parameters used in the generation of each synthetic sequence. Models stands for the number of finite-context models used. SI represents the context depth of the side information needed for representing the identification of the model used for each block.

Seq.	Models	SI	Block size	Size
A	1	-	-	1,000,000
B	4	5	100	1,000,000
C	8	5	100	1,000,000
D	8	0	100	1,000,000
E	8	1	100	1,000,000
F	8	3	100	1,000,000
G	8	5	100	1,000,000

Table 1 illustrates the generation parameters used to create these synthetic sequences.

3.1. Analysis of the information content

In order to compare the results between the previous method and the current one, we used the state-of-the-art XM encoder [19] to analyze the information content of the synthetic sequences, as can be seen in Fig. 2.

The synthetic sequence A was generated using a single finite-context model with context depth $k = 6$. The synthetic sequence B was generated using four competing finite-context models with context depths $k = 4, 8, 12, 16$. Sequence C was generated using eight competing finite-context models with context depths $k = 2, 4, 6, 8, 10, 12, 14, 16$. In Fig. 2, it is possible to observe that the synthetic sequence A is almost absent of low entropy valleys, and, therefore, it is very different from what we observe in real sequences. On the other hand, the synthetic sequences B and C, which have been generated with more than one model, have low entropy valleys, a characteristic more evident in sequence C. In fact, the number of models used to generate the sequences seems to be very important for increasing the number of low entropy valleys, rendering the synthetic sequences more similar to the real ones. This provides evidence to the conjecture that multiple competing finite-context models provide better generating results than single finite-context models.

Since the generator relies on the side information finite-context model for selecting the generating model that is used for each data block, we analysed how the produced sequences depend on this parameter (the SI column in Table 1). Observing Fig. 3, it seems that the low entropy valleys in the synthetic sequences are related to the side information context depth, showing that this parameter is important. In fact,

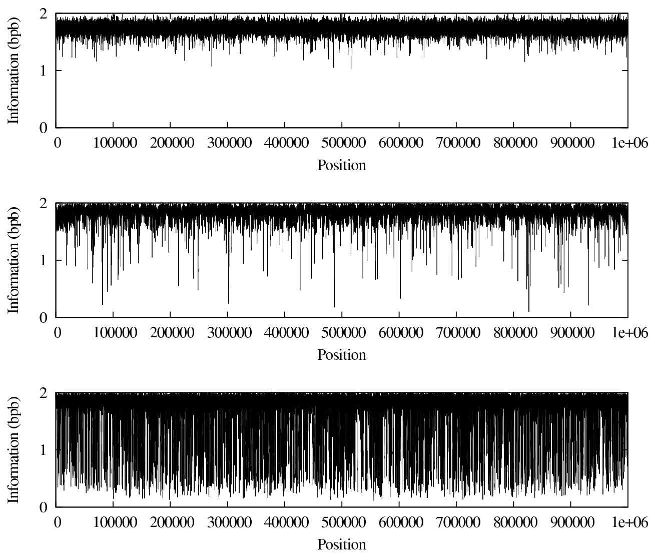

Fig. 2. Plots of the information content related to three synthetic sequences (A, B, C). The first row shows the information content of the sequence A, the second row of sequence B and the last row of sequence C. The sequences have been processed in both directions, low-pass filtered using averaging and a Blackman window of size 21, and combined according to the minimum value attained in each direction.

increasing the side information context depth provides more low entropy valleys in the information sequences. Given these results, we conclude that the approach based on multiple competing finite-context models provides better sequence representation, as can be verified through the observation of the information profiles.

3.2. Kullback-Leibler divergence

We used the Kullback-Leibler (KL) divergence to measure the difference between the probability distribution of the original sequences and the probability distribution of the corresponding synthetic sequences, according to different word sizes (w). Table 2 presents the values of this divergence for several values of w.

As can be seen, the divergence is small even for words of larger size, strengthening the idea that the synthetic sequences are statistically similar to the original sequences. It can also be observed that the divergence increases when the size of the sequence decreases. This is due to the fact that, when the depth of a model increases, the available original data might not be sufficient.

4. CONCLUSIONS

In a previous work [14], we have shown that finite-context modelling can be used for sequence generation. Also, it is known that DNA is better represented by multiple finite-context models [15] than by single models. For this reason, we have improved the generator, allowing the generation of sequences by multiple competing finite-context models.

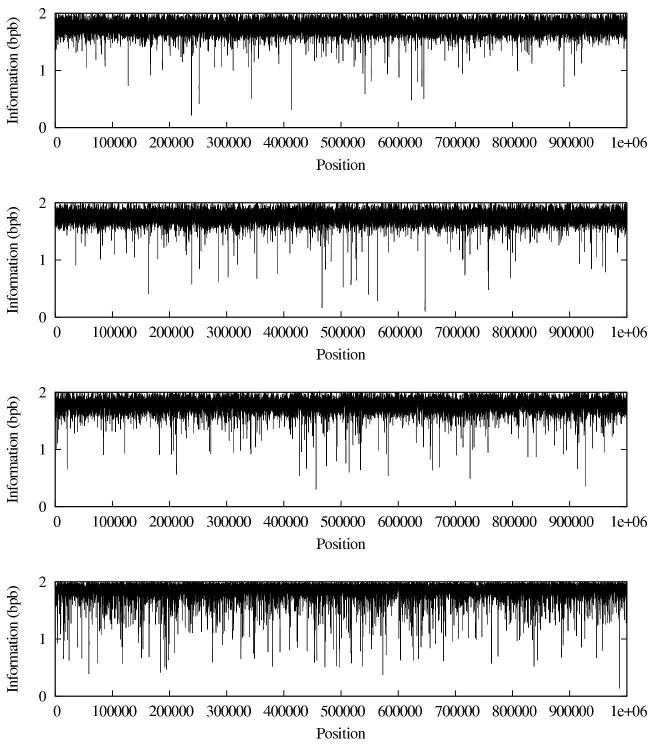

Fig. 3. Plots of the information content of the synthetic sequences (from top to bottom, D, E, F, G) based on the XM information content analysis.

We have studied the information content of the sequences produced by this generator, and we arrived at the conclusion that they present greater similarity to the real sequences, revealing a significant increase in the low entropy valleys, and attaining synthetic sequences that are statistically closer to the training sequences than those produced by the previous method, as shown by the Kullback-Leibler divergence distance.

5. ACKNOWLEDGMENT

This work was supported in part by the grant with the COMPETE reference FCOMP-01-0124-FEDER-010099 (FCT, Fundação para a Ciência e Tecnologia, reference PTDC/EIA-EIA/103099/2008).

6. REFERENCES

[1] S. W. Shin and S. M. Kim, "A new algorithm for detecting low-complexity regions in protein sequences,"

Table 2. Kullback-Leibler divergence, for several word sizes, w, between the original chromosomes and the synthetic chromosomes. “Chr” indicates the chromosome and “nBases” the number of bases.

Chr	nBases	w-2	w-3	w-4	w-5	w-6	w-7	w-8	w-9	w-10
1	218712898	0.0020	0.0041	0.0069	0.0108	0.0169	0.0259	0.0377	0.0522	0.0800
7	154546299	0.0064	0.0107	0.0157	0.0218	0.0306	0.0428	0.0586	0.0781	0.1228
21	33924742	0.0094	0.0153	0.0217	0.0292	0.0394	0.0537	0.0739	0.1135	0.3174

Bioinformatics, vol. 21, no. 2, pp. 160–170, 2005.

[2] A. Hartemink, Er J. Hartemink, D. Gifford, and J. Khodor, “Automated constraint-based nucleotide sequence selection for DNA computation,” in *4th Int. Meeting on DNA-Based Computing*, 1998, pp. 227–235.

[3] R. Penchovsky and J. Ackermann, “DNA library design for molecular computation,” *Journal of Computational Biology*, vol. 6, no. 2, pp. 215–229, 2003.

[4] A. Frutos, A. Thiel, A. Condon, L. Smith, and R. Corn, “DNA computing at surfaces: four base mismatch word design,” in *3rd DIMACS Workshop DNA Based Computers*, 1997, p. 238.

[5] M. Arita and S. Kobayashi, “DNA sequence design using templates,” *New Gen. Comput.*, vol. 20, pp. 263–277, July 2002.

[6] U. Feldkamp, S. Saghafi, W. Banzhaf, and H. Rauhe, “DNASequenceGenerator: a program for the construction of DNA sequences,” in *DNA Computing: 7th Int. Workshop on DNA-Based Computers, DNA7*, 2001, vol. 2340 of *LNCS*, pp. 23–32.

[7] F. Tanaka, M. Nakatsugawa, M. Yamamoto, T. Shiba, and A. Ohuchi, “Developing support system for sequence design in DNA computing,” in *Revised Papers from the 7th Int. Workshop on DNA Based Computers: DNA Computing*, 2002, pp. 129–137.

[8] A. Marathe, A. Condon, and R. Corn, “On combinatorial DNA word design,” *Journal of Computational Biology*, vol. 8, pp. 201–220, 1999.

[9] R. Deaton, J. Chen, H. Bi, M. Garzon, H. Rubin, and D. Wood, “A PCR based protocol for in vitro selection of noncrosshybridizing olgionucleotides,” in *Proc. of the 8th Int. Workshop DNA Based Computers*, 2002, pp. 196–204.

[10] C. Heitsch, A. Condon, and H. Hoos, “From RNA secondary structure to coding theory: A combinatorial approach,” in *Proc. of the Eighth Int. Meeting on DNA Based Computers*. 2002, LNCS, Springer-Verlag.

[11] R. Deaton, J. Chen, R. Murphy, J. Rose, D. Franceschetti, and S. Stevens, “Reliability and efficiency of a DNA-based computation,” *Pattern Recognition Letters*, vol. 80, no. 2, pp. 417–420, 1998.

[12] M. Arita, A. Nishikawa, M. Hagiva, K. Komiya, H. Gouzu, and K. Sakamoto, “Improving sequence design for DNA computing,” in *Proc. of Genetic and Evolutionary Computation Conf.*, 2000, pp. 875–882.

[13] S. Shin, D. Kim, I. Lee, and B. Zhang, “Evolutionary sequence generation for reliable DNA computing,” in *Proc. of the 2002 Congress on Evolutionary Computation, CEC’02*, May 2002, vol. 1, pp. 79–84.

[14] D. Pratas, A. J. Pinho, A. J. R. Neves, and C. A. C. Bastos, “DNA synthetic sequences generated by finite-context models,” in *Proc. of the 16th Portuguese Conf. on Pattern Recognition, RecPad 2010*, Oct. 2010.

[15] A. J. Pinho, A. J. R. Neves, D. A. Martins, C. A. C. Bastos, and P. J. S. G. Ferreira, “Finite-context models for DNA coding,” in *Signal Processing*, S. Miron, Ed., pp. 117–130. INTECH, Mar. 2010.

[16] A. J. Pinho, A. J. R. Neves, and P. J. S. G. Ferreira, “Inverted-repeats-aware finite-context models for DNA coding,” in *Proc. of the 16th European Signal Processing Conf., EUSIPCO-2008*, Lausanne, Switzerland, Aug. 2008.

[17] A. J. Pinho, A. J. R. Neves, C. A. C. Bastos, and P. J. S. G. Ferreira, “DNA coding using finite-context models and arithmetic coding,” in *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, ICASSP-2009*, Taipei, Taiwan, Apr. 2009.

[18] B. Haubold and T. Wiehe, “How repetitive are genomes?,” *BMC Bioinformatics*, vol. 7, no. 1, pp. 541, 2006.

[19] M. D. Cao, T. I. Dix, L. Allison, and C. Mears, “A simple statistical algorithm for biological sequence compression,” in *Proc. of the Data Compression Conf., DCC-2007*, Snowbird, Utah, 2007.