STORING DIGITAL DATA INTO DNA: A COMPARATIVE STUDY OF QUATERNARY CODE CONSTRUCTION

Melpomeni Dimopoulou, Marc Antonini, Pascal Barbry, Raja Appuswamy

STORING DIGITAL DATA INTO DNA: A COMPARATIVE STUDY OF QUATERNARY CODE CONSTRUCTION

Melpomeni Dimopoulou, Marc Antonini
Université Côte d'Azur
I3S, CNRS, UMR 7271
France

Pascal Barbry
Université Côte d'Azur
IPMC, CNRS, UMR 7275
France

Raja Appuswamy
Eurecom
Campus SophiaTech
France

ABSTRACT

The exponential increase of digital data that is being generated every year along with the capacity and durability limits of conventional storage devices are raising one of the greatest challenges for the field of data storage. The use of DNA for digital data archiving is a very promising alternative as the biological properties of the DNA molecule allow the storage of a huge amount of information into a very limited volume while also promising data longevity for centuries or even longer. In this paper we present a comparative study of our work with the state of the art solutions, and show that our solution is competitive.

Index Terms— DNA coding, data storage, quaternary code, sequencing noise robustness

1. INTRODUCTION

In the last decade several studies have been carried out on the use of DNA as a means of digital data storage to deal with the problem of storage capacity and durability. It is strongly believed that the DNA is a molecule appropriate for the efficient storage of digital data, which can outperform by far the existing storage means. More precisely it has been proven that DNA can store about 215 petabytes in a single gram of DNA. This is in line with the great chemical stability of the DNA molecule as long as it is protected from contacts with water and oxygen. Indeed, scientists have managed to decode the DNA of a woolly mammoth that had been trapped in permafrost for 39,000 years. DNA is complex molecule corresponding to a succession of four types of nucleotides (nts), Adenine (A), Thymine (T), Guanine (G), Cytosine (C). It is this quaternary genetic code that inspired the idea of DNA data storage which suggests that any digital information can be encoded into a DNA sequence of A, T, C, G. The main challenge lies in the restrictions imposed by the biological procedures of DNA synthesis (writing) and DNA sequencing (reading), which are involved in the encoding process and introduce significant error in the encoded sequence. Furthermore, the high expenses for DNA synthesis and sequencing yield the need for finding solutions for reducing the total cost for storing data into DNA. More precisely, it cost today $7000 to synthesize 2 megabytes of data into DNA, and another $2000 to read it by sequencing. Recent works tackle the problem of digital data storage onto DNA. For example, in [1] there has been a first attempt to store data into DNA while also providing a study of the main causes of biological error. In order to deal with errors previous works in [2] and [3] have suggested dividing the original file into overlapping segments so that each input bit is represented by multiple oligos. However, this procedure introduces extra redundancy and is poorly scalable. Other studies ([4],[5]) suggest the use of Reed-Solomon code in order to treat the erroneous sequences while in [6] a new robust method of encoding has been proposed to approach the Shannon capacity. Finally, latest works in [7] have introduced a clustering algorithm to provide a system of random access DNA data storage. Nevertheless, all of these approaches mainly try to transcode a bit stream onto a DNA sequence without taking into account the original input data characteristics. In [8] we have made a first attempt to build an efficient image coder for the storage of digital images into DNA. The proposed encoder uses the classical DNA data storage workflow used in the state of the art introducing an extra sub-part of image compression to allow control of the high DNA synthesis cost. Furthermore as the encoding algorithms suggested by previous works have some disadvantages when used along with image compression methods we have presented a new quaternary encoder which is suitable for the proposed workflow. In this work we present a more detailed theoretical study of the efficiency and the advantages of our encoding algorithm comparing it with the existing state of the art encoding methods for DNA data storage. For further information about the experimental details and results of our proposed solution, readers can refer to [8] where a wet lab experiment has been carried out.

2. GENERAL OVERVIEW

DNA data storage is a new emerging field of research which can be described by the general workflow of figure 1. The classical workflow is composed by five main sub-processes. The first part is the encoding of a sequence of digital symbols into a quaternary sequence of A, T, C and G. Then the encoded sequences need to be synthesized into DNA. As the DNA synthesis is error free under the constraint that the strands to be synthesized are no longer that 200nts, the encoded data has to be cut into smaller chuncks (oligos) using a formatting procedure in which special headers should also be added denoting the order of the information in the initial long sequence. Then the oligos are synthesized into DNA in vitro and stored into special capsules which prevent it from contacts with water and oxygen, promising reliable storage for hundreds of years. To read back the stored data we use some special machines that are called sequencers. A main drawback of the sequencing part is the fact that it is prone to errors and there are some particular ill-cases that produce high error rates. Those ill-cases can be avoided by respecting the following rules in the encoding of the DNA sequence:

- No homopolymer runs. Repetition of the same symbol more than 3 times should be avoided
- $\%A, T \leq \%G, C$. The content of G and C should not be greater than the percentage of as and Ts. A good ratio being 1.5 AT for 1 GC.
- No repetitions of short patterns.

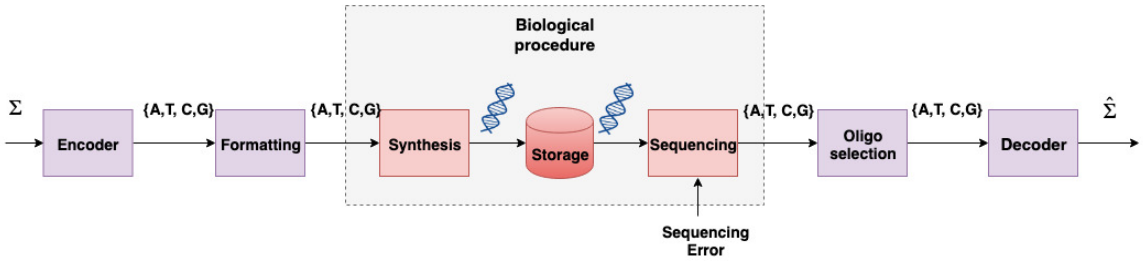

Fig. 1: General workflow of DNA data storage

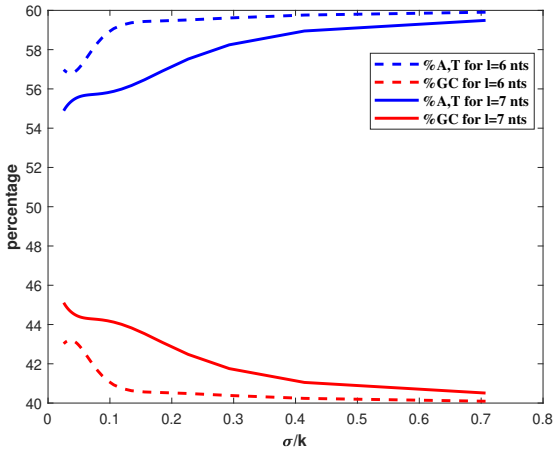

Fig. 2: Evolution of the percentages of A,T (blue curves) and G,C (red curves) for a centered Gaussian source of variance σ^2 and mean $\mu = \frac{k}{2}$ (k denoting the number of different source symbols) in function of the source dynamic normalized by k. The full line and dashed curves represent two different cases of codeword length l of 6 and 7 nucleotides respectively.

By avoiding the above cases the sequencing will be more reliable but not error free. The addition of extra redundancy using PCR amplification (a specific biological process) that uses a special enzyme to create many copies of the synthesized strands, may improve the reliability of the sequencing. Furthermore the sequencers themselves are adding extra redundancy as they are producing copies while reading the strands. For further information readers may refer to [9]. The sequencing results in many copies of the original synthesized oligos many of which will contain errors. Thus, one needs to select which oligos are the less corrupted. Under the hypothesis that the most frequent oligos will be the most representative ones, it is those sequences that will be selected for the decoding. In addition to this selection, error correction can further improve the quality of the reconstructed information. Finally using the inverse procedure of the one followed during the encoding, the initial information is retrieved. At this point it is important to point out the fact that this is the standard format of the digital data storage workflow and it is being used in the literature.

3. PROPOSED ENCODING SOLUTION

3.1. Code construction

The encoding algorithm proposed in [8] for the construction of the code $\mathcal{C}^*$ is guided by the first two restrictions of DNA data coding described in the previous section. The third and last restriction will be handled in the following section. The main idea is the creation of codewords from a set of duplets (pairs of symbols) which create an acceptable sequence when assembled in a longer strand. More precisely, the codewords are constructed by selecting elements from the following dictionaries:

- $\mathcal{C}_1 = \{AT, AC, AG, TA, TC, TG, CA, CT, GA, GT\}$
- $\mathcal{C}_2 = \{A, T, C, G\}$

Then, the code $\mathcal{C}^*$ is created by selecting elements from $\mathcal{C}_1$ and $\mathcal{C}_2$. Codewords of an even length l are constructed only by selecting $\frac{l}{2}$ pairs from dictionary $\mathcal{C}_1$. Codewords of an odd length are constructed by selecting $\frac{l-1}{2}$ pairs from $\mathcal{C}_1$ also adding a symbol from $\mathcal{C}_2$ at the end of the codeword. To ensure that the code does not create homopolymers, the dictionary $\mathcal{C}_1$ does not contain pairs of the same symbol. This means that the pairs AA, TT, CC, and GG are omitted as their consecutive repetition could create homopolymers. Furthermore, to keep the C and G percentage lower or equal to the one of A and T the pairs GC and CG are also not included. To verify this last claim we have computed the evolution of the A,T and G,C percentages created using our code for a source of symbols that follows a Gaussian distribution, in function of the variance σ^2 of the source. The result is illustrated in figure 2. As expected, we can see that the amount of GC content tends to 40% while the one of AT tends to 60% when the source becomes uniform (big values of σ).

3.2. Mapping

In order to encode a source sequence onto DNA we define the code Γ as the application: $\Gamma : \Sigma \rightarrow \mathcal{C}^\star$ where $\mathcal{C}^\star$ is a dictionary composed by $L \geq 2k$ codewords c_i of length l, and Σ the set of source symbols. We denote $\Gamma(s_i) = c_i$ the codeword associated with a value $s_i \in \Sigma$.

When the source symbol distribution to be encoded is not uniform, repetition of the same symbol can occur allowing to the creation of pattern repetition in the DNA code. The use of existing algorithms for the encoding of such a sequence into DNA would thus create pattern repetitions. In this work, in order to avoid those repetitions, we developed a new algorithm based on pseudorandom mapping which associates a source symbol to more than one possible codewords. By doing so, we ensure the representation of each symbol by at least two codewords so that in the case of repetition of the same symbol there will be more than one possible codewords that can be chosen so that no patterns are created. More precisely our algorithm maps the index i to the codewords of $\mathcal{C}^*$ as described in figure 3b. The code Γ is constructed so that each value in Σ is mapped to a set of different non-empty quaternary codewords in $\mathcal{C}^\star$ following a one-to-many relation in such a way that it is uniquely decodable. Since we ensure $L \geq 2k$, the pseudorandom mapping can at least provide two possible codewords for one input symbol. More precisely, the mapping is described by the following steps:

1. Compute the number of times m that k can be replicated into the total size L of the code $\mathcal{C}^*$: $m = \lfloor \frac{L}{k} \rfloor$,
2. The mapping of the value s_i to a codeword c_i is given by: $\Gamma(s_i) = \mathcal{C}^*(i + rand(0, m-1) * k)$.

3.3. Discussion

The mapping described in the previous section requires $L \geq 2k$ and thus ensures that every symbol in Σ must be represented at least twice in the code $\mathcal{C}^*$ to avoid pattern repetition. However, when $L \leq 2k$ (as in the case described by figure 3a), to exploit the wasted words in C^* we can perform a multiple assignment of the wasted codewords to the most frequent symbols in Σ. In the extreme case where $L \geq 2k$ each symbol will have at least two assigned codewords in C^*.

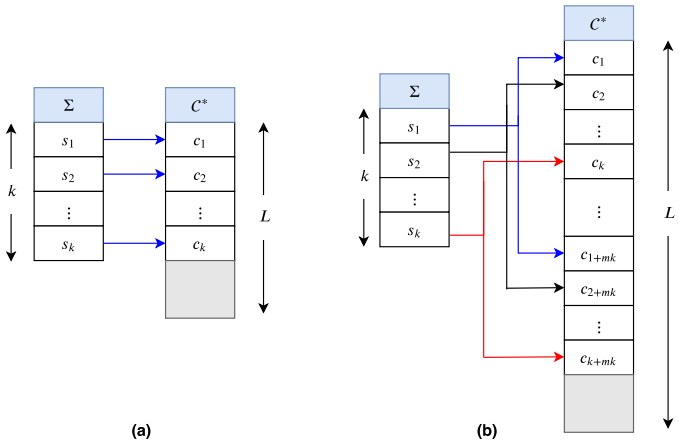

Fig. 3: Mapping symbols to codewords: (a) $k \leq L \leq 2k$ and (b) $L \geq 2k$. The gray area corresponds to unused codewords, if they exist.

4. COMPARISON TO THE STATE OF THE ART

DNA data storage is a new field of research which is expected to make a breakthrough in the domain of "cold" digital data archiving. As briefly described in section 1, some existing pioneering works suggest different algorithms for encoding the digital information into a quaternary sequence of A, T, C, G. In this section, we describe the advantages of the encoding algorithm proposed in this work in comparison to existing encoding methods. The first attempt of encoding digital data into DNA is described in [1] by the works of Church *et* al. In this work each binary bit is encoded to one nucleotide giving a total coding potential of 1 bit/nucleotide. To improve the coding potential as well as the robustness of the encoding to errors following works have adopted some more complicated encoding algorithms. More precisely Goldman et al. in [2], have proposed an algorithm that respects the constraint from section 2 of avoiding homopolymer runs to improve the quality of sequencing. This encoding applies a ternary huffman algorithm to compress the binary sequence into a ternary stream of three symbols (trits). Then, each of the trits is encoded into a symbol from the dictionary $\{A, T, C, G\}$ each time avoiding the symbol that has been previously used. However, unlike our proposed algorithm, in the encoding of a quantized or a sparse signal (as for instance the wavelet coefficients of a DWT transform where a same quantized value can be consecutively repeated many times), this encoding algorithm can create pattern repetitions which is an ill-case leading to a higher error probability at the phase of sequencing [10]. Another interesting work has been proposed by Blawat *et* al [5].

Codec	JPEG 2000				JPEG			
PSNR (dB)	57	48.1	40.2	35.9	61.5	49.2	40.6	35.6
Our rate (bits/nt) $k \leq L \leq 2k$	3.8	6.4	16	32	2.46	3.86	9.4	23.6
Our rate (bits/nt) $L \geq 2k$	2.7	5.3	13.3	26.6	2.05	3.21	7.7	19.72
Goldman et al. rate (bits/nt)	2.9	5.7	14.4	28.8	2.25	3.53	8.6	22.4

Table 1: Comparison of Our encoder with $L \geq 2k$ and our encoder with $k \leq L \leq 2k$ to Goldman et. al. [2] Here the rate is expressed in bits per nucleotide (bits/nt) to highlight the coding potential of the different solutions.

This encoding proposes the use of 5 nucleotides to encode 8 bits of information using a method for avoiding homopolymers. Furthermore the encoding inserts some randomization in the selection of the codewords which can be exploited for avoiding pattern repetitions as well as for correcting some types of errors that may occur. The coding potential of this method is 5 nucleotides per 8 bits of binary sequence which is equivalent to 1.6 bits/nt.To encode 8 bits (255 different symbols), our proposed algorithm also needs 5 nucleotides. Nevertheless, a strong advantage of our algorithm is the fact that it can be extended to the encoding of more than 8 bits of information and it can be applied to any type of input data (binary or not). In the works of Grass *et* al [4], the encoding is performed using Reed Solomon codes. This encoding achieves a coding potential of 1.187 bits/nt introducing redundancy in order to introduce error correction but similarly to [5] it is being applicable only in a binary stream. Bornholt *et* al in [3] have applied the same encoding as in [2] improving the encoding scheme and avoiding the fourfold redundancy which is suggested by the latter and synthesizes each DNA chunk in 4 shifted copies of the initial sequence. For further information about the fourfold redundancy the reader can refer to [2].

Finally, Erlich *et* al [6] have implemented an encodig using Fountain codes to reach a high coding potential. Similarly to most of the previously mentioned works, despite the efficiency in terms of information density, this type of encoding is only applicable to binary information while also being very expensive in computational cost. To evaluate the efficiency of our encoding algorithm we have compared it to the one proposed by [2]. The choice of this work for the comparison is for two main reasons. Firstly, the work proposed by [2] is one of the most popular ones and is to our knowledge the most widely used until this day. Secondly, similarly to our encoding, the algorithm proposed by this work can be applied to any type of symbols and is not limited to the encoding of binary data. For the comparison we have used the JPEG and JPEG2000 codecs to compress a set of 10 different images [1] of size 1510×5120 pixels to different compression rates. Each byte of the binary stream produced by JPEG and JP200 codecs represents a different symbol and thus $k = 2^8 = 256$. We then compare our algorithm to the one proposed in [2] to encode the binary stream into a quaternary sequence of A, T, C and G and have built the curves of coding potential (expressed in nts/pixel) in function of the PSNR. The results of this comparison are illustrated in figure 4. In table 1 we also show the coding rates expressed in bits/nt for different values of PSNR. More precisely, in our results we compare the encoder of [2] to the one proposed in this work for two different cases of mapping. The

[1] https://people.xiph.org/~tdaede/pcs2015_vp9_vs_x264/png/

Parameter	Church et al. [1]	Goldman et al. [2]	Grass et al.[4]	Bornholt et al.[3]	Blawat et al. [5]	Erlich et al.[6]	Our work (raw data)
Input data (Mbytes)	0.65	0.75	0.08	0.15	22	2.15	0.26
Coding potential (bits/nt)	1	1.58	1.78	1.58	1.6	1.98	1.6
Redundancy	1	4	1	1.5	1.13	1.07	1
Error correction	No	Yes	Yes	No	Yes	Yes	No

Table 2: Comparison to previous works

first case ($L \geq 2k$), is the mapping of one symbol to at least two different codewords as proposed in section 3.2, and the second case ($k \leq L \leq 2k$) corresponds to the mapping of the most frequent symbols to the codewords that are left unused as proposed in the discussion in 3.3. Those results reveal the fact that our proposed encoder's efficiency in terms of coding potential in the encoded stream of nucleotides is comparable to the encoder proposed by Goldman *et* al and even slightly better. It is also very interesting to point out again that thanks to mapping repetition our encoder ensures that the encoded sequence of nucleotides does not contain pattern repetitions which endanger the reliability of the sequencing and can therefore produce sequencing errors.

Furthermore, the encoding solution proposed in [2] embeds a ternary Huffman tree to transform the output bitstream of some codecs (like JPEG and JPEG2000) into a ternary sequence of 0,1 and 2 and then encodes it into a sequence of A, , C and G. Because of the use of Huffman, the probabilities of the different input symbols should be known or transmitted to the decoder as well. To the contrary, our proposed quaternary coder produces a simple fixed length code that doesn't require the transmission of any side information, allowing easier error correction in case of an insertion or deletion error. Hence, given also the fact that the proposed algorithm is flexible to modifications according to the encoding needs, those results are very encouraging while proposing a highly robust code. A comparison of the coding potential that has been reached using the different encoding approaches proposed by the state of the art are presented in table 2.

5. CONCLUSIONS

In this paper we have presented a further comparative study on a new robust encoder which has been proposed in our work in [8]. In this comparison, we have discussed the main advantages of our encoder including the fact that it can be applied on any type of source symbols and not only on binary data as in the case of most of the encoders proposed by previous works. Furthermore, our algorithm avoids creating pattern repetitions which can create more errors in the sequencing process. Finally we have proposed a comparison between our algorithm and the one of Goldman *et* al [2], by tanscoding the output bitstreams of JPEG and JPEG2000 encoded images into a quaternary sequence. This comparison has shown that the creation of a codebook using our proposed encoder can outperform the algorithm of Goldman *et* al. However our solution allows to deal with pattern repetition. Summing up all the assets of our proposed encoding, it is interesting to denote that such an algorithm would be appropriate to use for image coding: unlike previous works which use transcoding, it can be directly applied to the quantized coefficients of some transforms. In addition to this, it can also avoid the pattern repetitions that can occur due to the quantization. Consequently, this algorithm is a good candidate to be embedded to a workflow for image coding for DNA data storage.

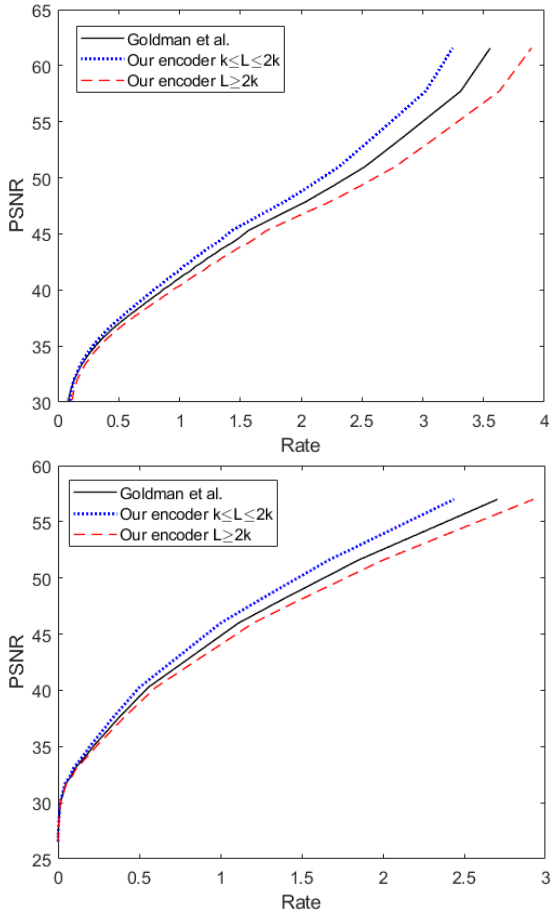

Fig. 4: Comparison to the encoding algorithm of Goldman *et.* al. For the encoding we consider each byte (8 bits, k=256) as a symbol to be encoded. The figures show the evolution of the PSNR (dB) in function of the Rate (nts/pixel) for different compression qualities using the JPEG codec (top figure) and the JPEG2000 (bottom figure).

6. REFERENCES

[1] George M Church, Yuan Gao, and Sriram Kosuri, "Next-generation digital information storage in DNA," Science, p. 1226355, 2012.

[2] Nick Goldman, Paul Bertone, Siyuan Chen, Christophe Dessimoz, Emily M LeProust, Botond Sipos, and Ewan Birney, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," Nature, vol. 494, no. 7435, pp. 77, 2013.

[3] James Bornholt, Randolph Lopez, Douglas M Carmean, Luis Ceze, Georg Seelig, and Karin Strauss, "A DNA-based archival storage system," ACM SIGOPS Operating Systems Review, vol. 50, no. 2, pp. 637–649, 2016.

[4] Robert N Grass, Reinhard Heckel, Michela Puddu, Daniela Paunescu, and Wendelin J Stark, "Robust chemical preservation of digital information on DNA in silica with error-correcting codes," Angewandte Chemie International Edition, vol. 54, no. 8, pp. 2552–2555, 2015.

[5] Meinolf Blawat, Klaus Gaedke, Ingo Huetter, Xiao-Ming Chen, Brian Turczyk, Samuel Inverso, Benjamin W Pruitt, and George M Church, "Forward error correction for DNA data storage," Procedia Computer Science, vol. 80, pp. 1011–1022, 2016.

[6] Yaniv Erlich and Dina Zielinski, "DNA fountain enables a robust and efficient storage architecture," Science, vol. 355, no. 6328, pp. 950–954, 2017.

[7] Lee Organick, Siena Dumas Ang, Yuan-Jyue Chen, Randolph Lopez, Sergey Yekhanin, Konstantin Makarychev, Miklos Z Racz, Govinda Kamath, Parikshit Gopalan, Bichlien Nguyen, et al., "Scaling up DNA data storage and random access retrieval," bioRxiv, p. 114553, 2017.

[8] Melpomeni Dimopoulou, Marc Antonini, Pascal Barbry, and Raja Appuswamy, "A biologically constrained encoding solution for long-term storage of images onto synthetic DNA," in EUSIPCO, 2019.

[9] I Illumina, "An introduction to next-generation sequencing technology," 2015.

[10] Todd J Treangen and Steven L Salzberg, "Repetitive DNA and next-generation sequencing: computational challenges and solutions," Nature Reviews Genetics, vol. 13, no. 1, pp. 36, 2012.