Analysis of the influence of errors in DNA-based image coding

Jorge Encinas Ramos, Davi Lazzarotto, Michela Testolina, Touradj Ebrahimi
Ecole Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
{jorge.encinasramos, davi.nachtigalllazzarotto, michela.testolina, touradj.ebrahimi}@epfl.ch

Abstract

In the last decade, DNA has been increasingly investigated as an alternative medium for cold data storage, presenting several advantages over standard hard drives such as a higher density, longer lifespan and lower energy consumption. However, such coding methods are limited by biochemical constraints that elevate the probability of errors being added to the coded nucleotides during synthesis, storage, and sequencing. Although such errors can be limited by carefully designing the produced strands, it is unfeasible to avoid them completely. In this paper, we explore the impact of naturally induced errors on the performance of a DNA-based image coding by means of realistic simulations, demonstrating that the quality of the decoded images is severely impacted. We also propose an error correction scheme based on Reed-Solomon codes and Blawat encoding, which successfully removes the produced artifacts.

Keywords

Image compression, DNA-based compression, error correction

1 Introduction

The amount of generated and stored data has been growing at an increasing rate, requiring the construction of more and more data storage centers. Digital data is usually represented as bits having binary values, and the majority of persistent data is stored in magnetic tapes and hard drives. Although current technology has been advancing to increase efficiency, the high rates of storage requirements pose a logistic and environmental challenge due to energy consumption. In this context, DNA has been proposed as a serious candidate for storing rarely accessed data. Contrary to typical binary systems, DNA is suitable to represent information in a quaternary basis through four different nucleotides: adenine (A), cytosine (C), guanine (G), and thymine (T). DNA coding has, in fact, the advantage of being capable of storing data with much less energy resources while offering much higher density, and if good conservation conditions are met, preservation periods in the order of hundreds of years.

However, this technology still suffers from unique drawbacks and is not yet ready to replace other data storage methods in all scenarios. In practical applications, DNA strands have to be synthesized and stored in low luminosity rooms with controlled temperature. In order to retrieve the original information, the molecules are sequenced in a costly process, making it expensive, time consuming, and impractical for data to be accessed multiple times. Moreover, the entire pipeline cannot be executed without undesirable errors such as deletion, insertion, or substitution of nucleotides. The synthesized strands are subject to biochemical constraints which, if they are not met, can either increase the error rate or produce unstable molecules. Research in the field reveals that, in order to reduce errors, the produced molecules should avoid repeated patterns and homopolymers, i.e. repeated sequences of the same nucleotides, as well as high proportions of C and G nucleotides. Although the real effect of naturally induced errors can only be tested with the real-world implementation of a full pipeline with synthesis and sequencing, a faster setup for testing coding mechanisms can be implemented using error simulators [1] that attempt to reproduce the outcomes of such natural processes.

Figure 1: *Left to right: original image, simulation with in vivo storage model, recovered corrected image*

In spite of all the reported challenges, several implementations of DNA-based coding have been reported in the last few years. In a first attempt to store digital information in DNA, [2] translated 0 and 1 to (A, C) and (T, G), respectively, with the goal of saving a 659-Kbyte digital book. Later, [3] produced the first coding mechanism avoiding the creation of homopolymers by translating data into a ternary basis and using a rotating dictionary to generate nucleotides. In particular, the previously encoded nucleotide was always excluded from the available options to represent the next symbol. Recent solutions incorporate mechanisms for error correction by including redundancy in the binary symbols prior to the translation to DNA, allowing to successfully retrieve the original encoded data even af-

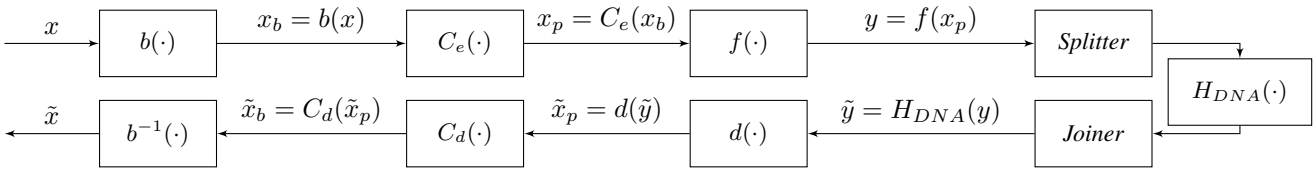

Figure 2: *Proposed workflow diagram*

ter errors being added to the nucleotide sequence, for example by using algorithms such as Reed-Solomon codes [4]. A number of works have also been devoted to developing methods to store image data into DNA. [5] leveraged the DCT from JPEG 1 and used Goldman with Huffman encoding to represent the obtained coefficients. Inspired by this solution, [6] proposed a transcoder to translate the coefficients of already compressed JPEG files into DNA. Both solutions assume a lossless DNA channel and do not take into consideration the errors induced by the storage pipeline. Recently, the usage of neural networks for designing effective and robust compression solutions is being explored. As an example, [7] proposed an image compression solution based on a learning-based convolution autoencoder that can be trained to be robust to substitution noise, which was nevertheless not evaluated against insertion or deletion errors. In this paper, the algorithm from [5] is used as a baseline by modeling the DNA channel with an error simulator [1], demonstrating that even a small percentage of errors causes severe degradation on the decoded image. A pipeline for error correction is then proposed and implemented, allowing for complete recovery of the original information from the distorted DNA strands.

2 Proposed error correction pipeline

The workflow proposed in this paper focuses on adapting existing Forward Error Correction (FEC) codes developed for binary usage to DNA applications. In order to achieve such a target, an Error Correction Block is wrapped around a binary pass-through stage, where the error correction code is applied before the information is returned to the DNA domain.

The proposed recoding procedure is shown in Figure 2. Let the unprotected source FASTA x first be converted into binary in a FASTA–binary conversion, labeled as $b(\cdot)$. This mapping may be as simple as a trivial fixed-length 2-bit mapping, where each base is assigned a fixed binary-represented number ranging from 0 to 3. This binary x_b is protected using the FEC code of choice via its encoding function, $C_e(\cdot)$. The protected binary x_p is then converted into a new FASTA $y = f(x_p) = f(C_e(b(x)))$, this time using a constraint-compliant encoding function $f(\cdot)$ that results in a DNA strand compliant with given biochemical constraints. The DNA sequence y is split into oligos of finite length (in our implementation, 200 nucleotides) and sent to the DNA channel H_{DNA}, which denotes all stages, i.e. synthesis, PCR, storage, and sequencing, of the DNA workflow and the subsequent errors introduced by them.

The DNA channel is here modeled using the MESA DNA error simulator [1], which reproduces the effects of all previously mentioned stages in individual oligos. For this reason, no identification methods such as barcodes were implemented to determine the position of oligos in the bitstream, which are concatenated after simulation to produce the FASTA $\tilde{y} = H_{DNA}(y)$. The protected binary equivalent inherently damaged by the DNA channel $\tilde{x}_p$ is then recovered using a conversion opposite to the one used in encoding, labeled as $d(\tilde{y})$. This binary-encoded information is then sent to the correction block $C_d(\cdot)$ resulting in the binary stream $\tilde{x}_b$. Here, if the redundancy available is enough to correct the errors in their binary form, the value of $\tilde{x}_b$ will be equal to x_b. Lastly, an operation inverse of the first mapping $b^{-1}(\cdot)$ is applied to reveal the estimated original FASTA $\tilde{x} = b^{-1}(C_d(d(\tilde{y})))$, which can be decoded with the corresponding coding algorithm.

The suitable error correction method must be selected considering the mappings and the relation between one nucleotide error and its corresponding binary error. In our simulations, the selected FEC code is a Reed-Solomon code in finite field GF(2^8), denoted as RS($255, 225, 31$). The choice for $f(\cdot)$ and $d(\cdot)$ was a simplified single-cluster Blawat encoding scheme [8], where every byte is converted into a 5-nucleotide tuple. Without clusters, it is impossible to locate the positions of added errors in $d(\cdot)$, but the encoding and decoding processes are less complex. The use of Blawat codes ensures that one substitution error alters at most two bits, that the overall GC content is balanced, as well as a maximum homopolymer length of 3 is reached. Moreover, this approach effectively turns insertions and deletions into additional substitution errors, resulting in a need for increased redundancy at the cost of equal processing of errors.

Since the length of the compressed FASTA in nucleotides is not necessarily an integer multiple of the block code size k, a padding sequence is added to the RS symbols. This padding is later scanned in the decoding process correlating it with the decoded sequence to locate and remove it. This way, a FASTA file with any length can be used with any code size without need for truncation.

The proposed method has the advantage that redundancy can be introduced without constraints, as the FEC code can be applied directly to the binary translation of the input FASTA and then split into oligos after re-encoding to DNA. This splitting procedure is a necessary step due to the DNA

synthesis constraints, which limit the synthesizable oligos to a few hundred nucleotides. This provides higher flexibility, as a different and further-optimized error correction algorithm may be used in this scheme, providing an extensive test bed of codes.

The designed scheme also provides effective protection to FASTA-encoded information that did not originally incorporate a correction scheme. This implies that the recoding mechanism ensures that the protected FASTA complies with the DNA-specific channel constraints independently of the source.

Finally, the scheme may offer the possibility of skipping the first encoding step, i.e. conversion from FASTA to binary, directly feeding binary information as input, with the goal of converting and protecting it for storage in DNA as a general purpose protect-encode block, similar to most existing DNA codec implementations.

3 Results and analysis

In the proposed workflow, the two steps allowing for an increase in the number of nucleotides in the output FASTA y when compared to x are the binary-FASTA conversions and the Reed-Solomon codes. Since $b(\cdot)$ and $f(\cdot)$ convert bits to nucleotides at different rates, increasing the amount of nucleotides by a factor of $5/4$, and since $C_e(\cdot)$ produces n symbols for every block of length k, the total ratio between stored nucleotides and source nucleotides equals to:

$$R_{rec} = \frac{5}{4} \cdot \frac{n}{k} = 1.417 \tag{1}$$

In other words, the proposed error correction method induces a nucleotide rate increase of approximately 41.7%. Without any detection of error position, the number of substitution errors that can be corrected on the GF(2^8) symbols from $\tilde{x}_p$ is:

$$\lfloor \frac{n-k}{2} \rfloor = 15 \tag{2}$$

Since all erasure and insertion errors in $\tilde{y}$ are converted into additional substitution errors when applying the Blawat decoding function $d(\cdot)$, it is possible to use the result from 2 to determine the maximum amount of errors added to the DNA channel that can be corrected by the proposed scheme. Considering that each block in x_p is composed of $n = 255$ GF(2^8) symbols, and each symbol is translated into 5 nucleotides by $f(\cdot)$, under the assumption that errors are sparse and uniformly distributed, then the maximum tolerated error rate is equal to $15/(255 * 5) = 1.18\%$.

The reported values depend on the level of redundancy added by the Reed-Solomon codes. Using a higher amount of redundancy symbols would allow the correction of more errors at the expense of increasing the nucleotide rate. As this work does not aim at reporting the optimal level of redundancy, and as DNA technology continues to advance, the appropriate level required may be chosen based on the assumptions about the DNA channel. For example, under assumptions of errorless synthesis and greatly improved sequencing or storage technologies, the number of redundancy symbols can be decreased as necessary to improve the code rate.

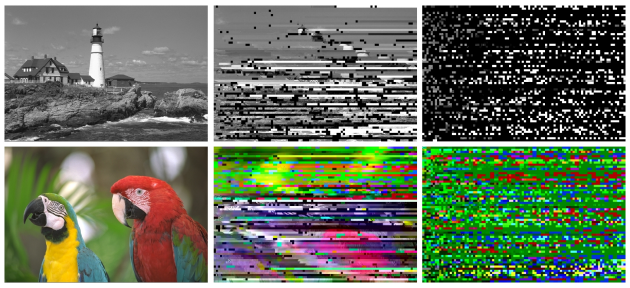

Figure 3: *Left to right: Original image, simulation with in vivo configuration, simulation with in vitro configuration*

The protection scheme described in the previous section was used in conjunction with the JPEG DNA Benchmark Codec [5] as the baseline producing DNA strands from still uncompressed images. Using two images from the Kodak dataset [9] as a test set, the corresponding FASTA files were obtained from the baseline. The error simulator was first applied directly into these sequences without any added protection to evaluate its effect on the decoded image. The results for two images of the test set can be observed in Figure 3.

The original undistorted images are reported in the left column, while the decoded images after error simulation uisng two distinct sets of configuration parameters without protection are reported in the middle and right columns. The different configurations can be obtained by selecting different equipment or technologies used in the DNA-domain workflow such as polymerases, storage hosts (*in vitro* or *in vivo*), or sequencers. The middle column shows the simulation results under a configuration based on an *in vivo* storage model using Escherichia Coli bacteria as the host. This results in a low number of erasures and insertions, and errors are mostly due to substitution. The right-most column contains the decoded results for simulation with *in vitro*-based storage, with error probability set to 0.5%, producing a high number of erasures, and a higher number of errors overall. Results show heavy degradation on the obtained visual result are inflicted because of the simulation. Although using an *in vivo* storage model results in lower loss on information, the final result is anyway wildly different from the original image. These results reveal that successful retrieval of media information without a strong correction scheme is not feasible.

In order to test the efficacy of the proposed error correction scheme, a crop of size 432x432 from test image *kodim-23* was selected and compressed into DNA with the baseline. The generated unprotected FASTA was then encoded using the error correction scheme. Both FASTA files were then served as input to the simulator, using an *in vivo* storage model configuration. Finally, they were both decoded back

Description	MS-SSIM
Unprotected (*in vivo* simulation)	0.0869
Protected (*in vivo* simulation)	0.9937
No simulation	0.9937

Table 1: *MS-SSIM scores for different configurations*

Description	FASTA length [nt]	Nucleotide rate [nt/px]
Unprotected (*in vivo*)	127,248	0.6818
Protected (*in vivo*)	181,050	0.9701

Table 2: *Nucleotide rate for different configurations*

to the image domain. Figure 1 depicts both retrieved of images. Visually, the image recovered from the protected FASTA doesn't present any difference from the original, while the unprotected FASTA produced an image with high visual distortion.

To quantify the obtained results, the MS-SSIM [10] quality metric was computed on a grayscale colormap of the images shown in Figure 1. The results are presented in Table 1.

The objective quality scores show that the distortions introduced by the error simulation have very high impact on the entire workflow, while the objective quality scores with error correction are identical to the image without any error simulation. Therefore, the error correction mechanism allowed to retrieve the same FASTA file as prior to the error simulation, effectively neutralizing the effect of the DNA channel.

Let us consider the bitrate with and without the protection scheme. The results can be seen in Table 2. The redundancy added considerably increases the number of nucleotides per pixel. The FASTA lengths for the protected image are computed before entering the DNA channel, after both encoding steps. This analysis reveals that a bitrate increase of around 42% was obtained, which is inline with the results from Equation 1.

4 Conclusions

In this paper, we presented an efficient pipeline that allows for correction of errors introduced by the synthesis and storage of DNA. Notably, the redundancy level analyzed in this paper allows for robustness against errors with an average bitrate increase, measured in nucleotides per pixels, of approximately 42%. Moreover, the proposed pipeline allows for a simple and flexible adaptation to error correction methods with increased performance. In further work, the optimal level of redundancy in different scenarios can be investigated.

Acknowledgments

The authors would like to acknowledge support from the Swiss National Scientific Research project entitled "Compression of Visual information for Humans and Machines (CoViHM)" under grant number 200020_207918.

References

[1] M. Schwarz, M. Welzel, T. Kabdullayeva, A. Becker, B. Freisleben, et D. Heider. Mesa: automated assessment of synthetic dna fragments and simulation of dna synthesis, storage, sequencing and pcr errors. *Bioinformatics (Oxford, England)*, 36(11):3322–3326, 2020.

[2] George M Church, Yuan Gao, et Sriram Kosuri. Next-generation digital information storage in dna. *Science*, 337(6102):1628–1628, 2012.

[3] Nick Goldman, Paul Bertone, Siyuan Chen, Christophe Dessimoz, Emily M LeProust, Botond Sipos, et Ewan Birney. Towards practical, high-capacity, low-maintenance information storage in synthesized dna. *nature*, 494(7435):77–80, 2013.

[4] Robert N Grass, Reinhard Heckel, Michela Puddu, Daniela Paunescu, et Wendelin J Stark. Robust chemical preservation of digital information on dna in silica with error-correcting codes. *Angewandte Chemie International Edition*, 54(8):2552–2555, 2015.

[5] Melpomeni Dimopoulou, Eva Gil San Antonio, et Marc Antonini. A jpeg-based image coding solution for data storage on dna. Dans *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 786–790. IEEE, 2021.

[6] Luka Secilmis, Michela Testolina, Davi Lazzarotto, et Touradj Ebrahimi. Towards effective visual information storage on dna support. Dans *Applications of Digital Image Processing XLV*, volume 12226, pages 29–35. SPIE, 2022.

[7] Xavier Pic et Marc Antonini. Image storage on synthetic dna using autoencoders. *arXiv preprint arXiv:2203.09981*, 2022.

[8] Meinolf Blawat, Klaus Gaedke, Ingo Hütter, Xiao-Ming Chen, Brian Turczyk, Samuel Inverso, Benjamin W. Pruitt, et George M. Church. Forward error correction for dna data storage. *Procedia Computer Science*, 80:1011–1022, 2016.

[9] Kodak Lossless True Color Image Suite (PhotoCD PCD0992), accessed: 13.03.2023. "`http://r0k.us/graphics/kodak/`".

[10] Zhou Wang, Eero P Simoncelli, et Alan C Bovik. Multiscale structural similarity for image quality assessment. Dans *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.