Simulating Noisy Channels in DNA Storage

Mayank Keoliya
National University of Singapore
mayankk@comp.nus.edu.sg

Puru Sharma
National University of Singapore
puru17@comp.nus.edu.sg

Djordje Jevdjic
National University of Singapore
jevdjic@comp.nus.edu.sg

***Abstract*—Compared to conventional storage mediums, DNA-based data storage offers benefits such as durability, high density and low energy consumption. With increased demand for DNA data storage, it has become important to quickly evaluate proposed approaches. However, experiments that involve reading and writing synthetic DNA are costly and time-consuming, thus requiring cheap and fast simulation prior to experimentation. DNA sequencing technologies such as Nanopore and Illumina have highly characteristic error profiles, and simulating them is challenging. We propose a DNA simulator for Nanopore data that improves on existing simulators by incorporating key parameters; our simulator better converges to error profiles of real data on most parameters.**

We show that the spatial distribution of errors within a strand is a key determinant of trace reconstruction accuracy; which is a factor that had not been considered by existing simulators.

***Index Terms*—DNA storage, noise simulation, information retrieval**

I. Introduction

DNA-based data storage involves the storage of data in the form of long sequences of synthetic DNA molecules known as *strands*. The DNA alphabet has four letters (or *bases*): A, G, C and T, and strands are typically 100-200 bases long. The original strands to be written (or *synthesized*) are called *reference strands*, which undergo errors, and are read (or *sequenced*) to yield several noisy copies called *reads*.

Key to designing synthetic DNA simulators is the problem of modeling the noisy channels in DNA storage. [1] proposed that errors should be modelled as a noisy Insertion-Deletion-Substitution (IDS) channel which accepts M DNA strands as input, injects noise via the IDS channel, and then samples N strands independently. The ratio $M : N$ is known as *sequencing coverage*, i.e. roughly the average number of reads per every reference strand. The reference strand is then recovered (with some error) by passing the noisy reads to *trace reconstruction algorithms*, such as BMA Look-Ahead [2] and Iterative Reconstruction [3].

II. Evaluation of Existing Simulators

Currently, only one prior work has attempted to devise an end-to-end simulator for DNA storage *viz.* DNASimulator [4]. Given a list of reference strands, a coverage N, and a DNA sequencing technology, it generates n noisy reads per reference strand. DNASimulator uses the following parameters for noisy channel: (i) probabilities of insertion, deletion and substitution (IDS) $(p_{ins}, p_{del}, p_{sub})$, (ii) base-specific conditional probabilities for IDS $(p_{(ins|A)}, ..., p_{(del|G)}, ...p_{(sub|T)})$, and (iii) probabilities for burst deletions $(p_{burst-dels})$. *Burst deletions* are consecutive deletions with length greater than one.

The key use of DNA simulators is to generate better error-encoding schemes and trace reconstruction algorithms to improve the per-strand and per-character accuracy of reconstructed strands. Thus, a suitable metric to evaluate the efficacy of DNA simulators is the difference between the post-reconstruction per-strand and per-character accuracies of real versus simulated data when both are passed to a suite of reconstruction algorithms. Convergence in per-strand and per-character accuracy indicates that the real and simulated data induce the same error types and localizations after reconstruction. Two state-of-the-art algorithms are used in this report: BMA Look-Ahead [2], and Iterative Reconstruction [3].

We evaluate the proposed metric at a low coverage ($N = 5$), since reconstruction accuracies are more sensitive to perturbations at low coverages [5] [6], and since DNA storage systems target low coverage to minimize sequencing cost. We analyzed per-strand and per-character accuracies of real Nanopore data versus data simulated by DNASimulator (Table I, rows 1-3). Both per-strand and per-character accuracy of simulated datasets were greater than those for real data. This demonstrates that DNASimulator is not adequate for simulating DNA storage. It makes the key assumption that the error rate is assumed to be independent of (i) a molecule's position in a strand, and (ii) type of error. As we demonstrate later, this assumption does not hold, and we can improve the similarity of simulated data by considering additional parameters in our simulation model.

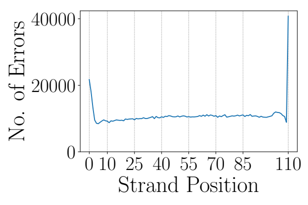
III. Proposed Simulation Model

In this section, we investigate the properties of the largest public Nanopore dataset released by Microsoft [7], and consider *skewed distribution* of errors as a key parameter.

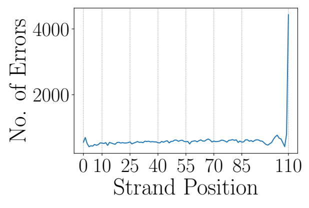
A. Analysis of Nanopore Data

The Nanopore dataset provided in [7] contains 10,000 reference strands, average coverage $\bar{N} = 26.97$, strand length $L = 110$, and $p_{ins} + p_{del} + p_{sub} = 5.9\%$.

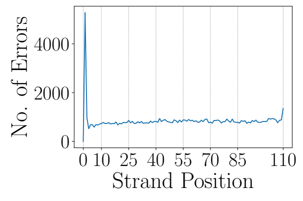
We compared the reference strands with the noisy reads by obtaining a histogram of (i) Hamming errors and (ii) gestalt-aligned errors at each strand position. The Hamming error is obtained by performing a base-wise equality check on the reference strand and each noisy copy; if the bases at a given position are not equal, then we increment the Hamming error at that position by 1. The gestalt-aligned comparison first runs the Ratcliff-Obershelp algorithm (also known as gestalt pattern



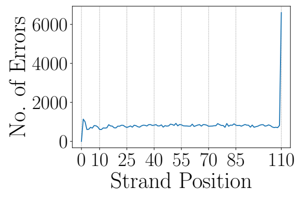
(a) Gestalt-aligned errors



(b) Insertion of G

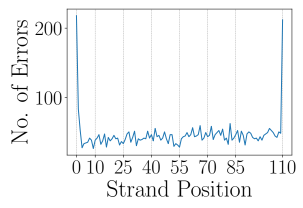


(c) Deletion of C

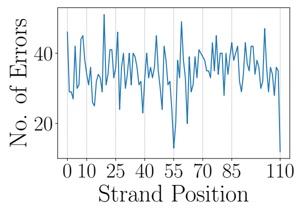


(d) Deletion of G

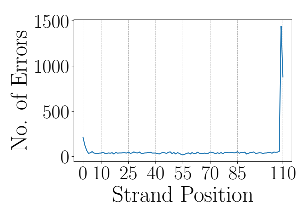
Fig. 1: Error distribution for Nanopore dataset



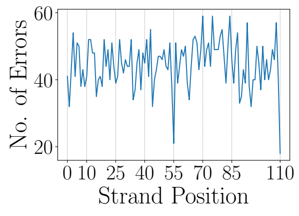
(a) Real



(b) Naive Simulator



(c) DNASimulator & Skew (Ours)



(d) DNASimulator [4]

Fig. 2: Gestalt-aligned errors after Iterative Reconstruction at Coverage = 5

matching) [8] to align the matching blocks first, and then count the number of errors *beginning* at a strand position. For example, consider a reference strand `ATGTG` and a noisy copy strand `AGTG`, where the second letter `T` has been deleted. The Hamming error histogram is `[0,1,1,1,1]`, the gestalt-aligned histogram is `[0,1,0,0,0]`, and the matching blocks are `A` and `GTG`.

We used a naive simulator as a baseline; only the aggregate probabilites for IDS ($p_{ins}, p_{dels}, p_{subs}$) were used to simulate data. We obtained the values of the parameters used in DNASimulator for the Microsoft Nanopore dataset by computing the edit distance operations between the reference strands and the noisy reads. For example, given a reference strand `ATGTG` and a noisy copy strand `AGTG`, the edit distance operations are `[=, del, =, =, =]`, yielding $p_{del} = 1/5 = 0.2, p_{(del|T)} = 0.5, p_{(del|A,G,C)} = 0$.

B. Skewed Distribution of Errors

We computed the gestalt-aligned histogram for Nanopore data (Fig. 1a). The histogram indicates that most of the errors occur at the beginning (positions 0 - 2) and end (position 109 = $L-1$) of the strand; 90% of errors originate at these positions. Prior studies [5] and [9] empirically validate the same for Illumina data. We observed that the probability of errors at terminal positions was four times the baseline at intermediate positions.

We inserted the skew for the aggregate errors (denoted as Skew$_1$) and computed the corresponding metrics for the simulated data. Fig. 2 provides a comparison of the gestalt-aligned errors for real data and simulations. Note that the histogram for skewed simulations fits more closely to that for real Nanopore compared to DNASimulator and naive simulation. Further, the per-strand and per-character accuracy converges for BMA, but diverges (excess decrease) for Iterative (Table I, row 4). This indicates that the Iterative algorithm is highly sensitive to skew at terminal positions.

TABLE I: Per-Strand and Per-Character Post-Reconstruction Accuracy (%) at Coverage = 5

Data	**BMA**		**Iterative**	
	Strand	**Char**	**Strand**	**Char**
Real Nanopore	29.04	87.74	66.70	90.32
Naive Simulator	68.21	93.45	90.60	99.31
DNASimulator (DS)	59.65	91.39	92.20	99.35
DS & Skew$_1$	**47.86**	**89.49**	**35.36**	**82.15**
DS & Skew$_1$ & Skew$_2$	**44.78**	**88.67**	**33.87**	**77.39**

C. Second-Order Errors

Second-order errors are specific error types e.g. insertion of `T`, substitution of `AA` for `A`, and so on. We find that along with a skew for the total number of errors (i.e. regardless of error type), there is a skew toward terminal positions for particular error types as well. Fig. 1 shows the skew for the three most common second-order types: insertion of G (1b), deletion of C(1c) and deletion of G (1d).

We inserted corresponding skews for the ten most common second errors into our model (denoted as Skew$_2$), simulated data, and then ran BMA and Iterative trace reconstruction. As for the previous case, the accuracy further converges for BMA, but diverges for the Iterative algorithm (Table I, row 5).

IV. Discussion

In this work, we devised and refined a simulator for noisy channels in DNA storage, and improved on DNASimulator, an existing simulator. We analyzed suitable metrics, datasets, and parameters for modelling the simulator. Compared to DNASimulator, our simulator converged closer to real data based on per-strand accuracy (15% v/s 38% difference for DNASimulator) and per-character accuracy (1% v/s 6% difference for DNASimulator) for the BMA algorithm. However, like DNASimulator, our simulator did not adequately converge for the Iterative algorithm.

There remain several limitations and challenges to designing DNA simulators. A key limitation is that existing simulators are not capable of generating intermediate clusters at different stages of the DNA storage pipeline; only the final noisy reads after sequencing are generated. Another challenge is the difficulty in choice of metric for the simulator; it is not obvious which trace reconstruction algorithm(s) or coverage(s) should be prioritized, since simulators might behave differently for different combinations. With the increase in number and type of DNA sequencing technologies, it is also difficult to analyze and simulate all possible noisy channels. Designing a generalizable set of parameters to describe all present DNA sequencing technologies remains an open problem.

References

[1] R. Heckel, G. Mikutis, and R. N. Grass, "A Characterization of the DNA Data Storage Channel," *Scientific Reports*, vol. 9, no. 9663, 2019. [Online]. Available: https://doi.org/10.1038/s41598-019-45832-6

[2] T. Batu, S. Kannan, S. Khanna, and A. McGregor, "Reconstructing strings from random traces." New Orleans, Louisiana, USA: Society for Industrial and Applied Mathematics, 2004, p. 910–918.

[3] O. Sabary, A. Yucovich, G. Shapira, and E. Yaakobi. (2020) Reconstruction Algorithms for DNA-Storage Systems. bioRxiv. [Online]. Available: https://www.biorxiv.org/content/early/2020/09/17/2020.09.16.300186

[4] G. Chaykin, N. Furman, O. Sabary, and E. Yakobi. (2020) Dna storage simulator. [Online]. Available: https://github.com/oyerush/DNASimulator

[5] T. P. Niedringhaus, D. Milanova, M. B. Kerby, M. P. Snyder, and A. E. Barron, "Random access in large-scale DNA data storage," *Anal Chem*, vol. 83, no. 12, pp. 4327–4341, 2011. [Online]. Available: https://doi.org/10.1021/ac2010857

[6] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, "A DNA-Based Archival Storage System," *SIGARCH Comput. Archit. News*, vol. 44, no. 2, p. 637–649, Mar. 2016. [Online]. Available: https://doi.org/10.1145/2980024.2872397

[7] R. S. Srinivasavaradhan, G. Sivakanth, H. D. Pfister, and S. Yekhanin, "Trellis bma: Coded trace reconstruction on ids channels for dna storage," in *2021 IEEE International Symposium on Information Theory (ISIT)*, 2021, pp. 2453–2458.

[8] J. W. Ratcliff and D. Metzener, "Pattern matching: The gestalt approach," *Dr. Dobbs Journal*, vol. 7, p. 46, 1988.

[9] L. Ceze, J. Nivala, and K. Strauss, "Molecular digital data storage using DNA," *Nature Reviews Genetics*, vol. 20, pp. 456–466, 2019. [Online]. Available: https://doi.org/10.1038/s41576-019-0125-3