Reconstruction Algorithms for DNA-Storage Systems

Omer Sabary[1,2], **Alexander Yucovich**[1], **Guy Shapira**[1], and **Eitan Yaakobi**[1]

[1]The Henry and Marilyn Taub Faculty of Computer Science, Technion, Haifa, 3200003, Israel.
[2]Electrical and Computer Engineering Dept., University of California, San Diego, La Jolla, CA 92093, USA.

Abstract—**In the *trace reconstruction problem* a length-n string $\boldsymbol{x}$ yields a collection of noisy copies, called *traces*, $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t$ where each $\boldsymbol{y}_i$ is independently obtained from $\boldsymbol{x}$ by passing through a *deletion channel*, which deletes every symbol with some fixed probability. The main goal under this paradigm is to determine the required minimum number of i.i.d traces in order to reconstruct $\boldsymbol{x}$ with high probability. The trace reconstruction problem can be extended to the model where each trace is a result of $\boldsymbol{x}$ passing through a *deletion-insertion-substitution channel*, which introduces also insertions and substitutions. Motivated by the storage channel of DNA, this work is focused on another variation of the trace reconstruction problem, which is referred by the *DNA reconstruction problem*. A *DNA reconstruction algorithm* is a mapping $R : (\Sigma_q^*)^t \to \Sigma_q^*$ which receives t traces $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t$ as an input and produces $\widehat{\boldsymbol{x}}$, an estimation of $\boldsymbol{x}$. The goal in the DNA reconstruction problem is to minimize the edit distance $d_e(\boldsymbol{x}, \widehat{\boldsymbol{x}})$ between the original string and the algorithm's estimation. For the deletion channel case, the problem is referred by the *deletion DNA reconstruction problem* and the goal is to minimize the Levenshtein distance $d_L(\boldsymbol{x}, \widehat{\boldsymbol{x}})$. In this work, we present several new algorithms for these reconstruction problems. Our algorithms look globally on the entire sequence of the traces and use dynamic programming algorithms, which are used for the *shortest common supersequence* and the *longest common subsequence* problems, in order to decode the original sequence. Our algorithms do not require any limitations on the input and the number of traces, and more than that, they perform well even for error probabilities as high as 0.27. The algorithms have been tested on simulated data as well as on data from previous DNA experiments and are shown to outperform all previous algorithms.**

I. Introduction

Recent years presented significant improvements and new methods for the technologies of DNA synthesis and DNA sequencing. This progress also introduced the development of data storage technology based upon DNA molecules. A DNA storage system consists of three important components. The first is the *DNA synthesis* which produces the *oligonucleotides*, also called *strands*, that encode the data. The second part is a storage container with compartments which stores the DNA strands, however without order. Finally, *sequencing* is performed to read back a representation of the strands, which are called *reads*. The processes of synthesizing, storing, sequencing, and handling strands are all error prone. Each step in these processes can independently introduce a significant number of errors, mostly of three types deletion, insertion, and substitution.

Current synthesis technologies are not able to generate a single copy for each DNA strand, but only multiple copies where the number of copies is in the order of thousands to millions. Hence, every strand has multiple copies and several of them are read during sequencing.

The encoding and decoding stages are two processes, external to the storage system, that convert the user's binary data into strands of DNA such that, even in the presence of errors, it will be possible to revert back and recover the original binary data. These two stages consist of three steps, which we refer by 1. *clustering*, 2. *reconstruction*, and finally 3. *error correction*. After the strands are read back by sequencing, the first task is to partition them into *clusters* such that all strands in the same cluster originated from the same synthesized strand. Then, the goal is to reconstruct each strand based upon all its noisy copies, and this stage is the main problem studied in this paper. Lastly, errors which were not corrected by the previous steps, should be corrected by the use of an error-correcting code.

The decoder's reconstruction algorithm is performed on each cluster to recover the original strand from the noisy copies in the cluster. In fact, this setup falls under the general framework of the *sequence reconstruction problem* which was first studied by Levenshtein [5] and the *trace reconstruction problem* [1]. In general, these models assume that the information is transmitted over multiple noisy channels, and the decoder, which observes all channel estimations, uses this inherited redundancy in order to correct the errors.

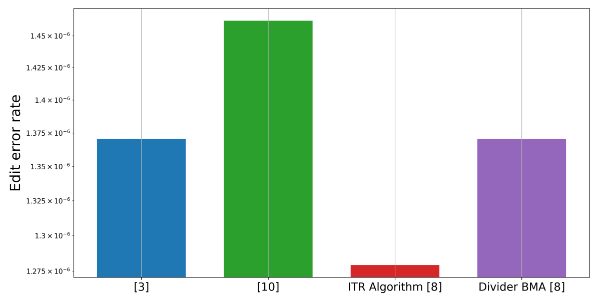
The main goal of the sequence reconstruction and the trace reconstruction problems is to find the required minimum number of channels in order to guarantee successful decoding either in the worst case or with high probability. However, in DNA-based storage systems we cannot control on the number of strands in each cluster. Hence, this work is focused on a variation of the trace reconstruction problem, which is referred by the *DNA reconstruction problem*. The setup is similar to the trace reconstruction problem. A length-n string $\boldsymbol{x}$ is transmitted t times over the *deletion-insertion-substitution channel* and generates t noisy copies called traces $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t$. A *DNA reconstruction algorithm* is a mapping $R : (\Sigma_q^*)^t \to \Sigma_q^*$ which receives the t traces $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t$ as an input and produces $\widehat{\boldsymbol{x}}$, an estimation of $\boldsymbol{x}$. The goal in the DNA reconstruction problem is to minimize $d_e(\boldsymbol{x}, \widehat{\boldsymbol{x}})$, i.e., the edit distance between the original string and the algorithm's estimation.

In our model we assume that the clustering step has been done successfully and present algorithms that work with a flexible number of copies and various probabilities for deletion, insertion, and substitution errors. We also apply our algorithms on data from previously published DNA-storage experiments [2], [4], [6] and compare our accuracy and performance with state of the art algorithms known from the literature.

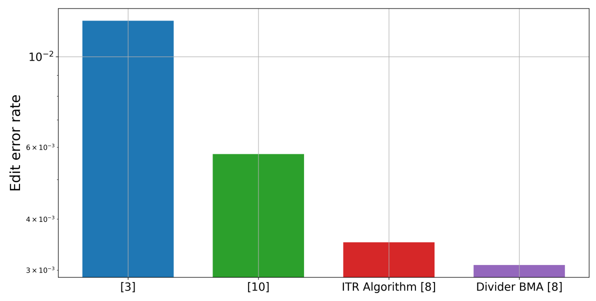
A comprehensive summary of the previous works in DNA-storage and their error charaterization, as well as the previous work regarding reconstruction problem and their application to DNA storage can be found in our full paper [8].

II. Our algorithms

In this work we present several reconstruction algorithms, designed for DNA storage systems. Since our purpose is to solve the reconstruction problem as it is reflected in DNA-storage systems, our algorithms aim to minimize the distance between the output and the original strands. The algorithms in this work are different from most of the previously published reconstruction algorithms in several respects. First, we do not require any assumption on the input. Second, our algorithms

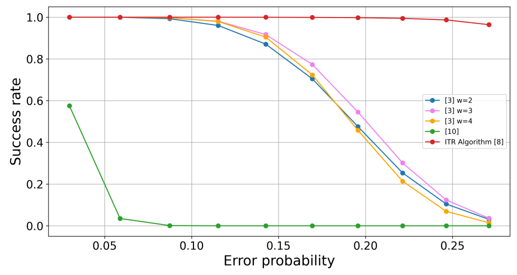


(a) Erlich and Zielinski [2].

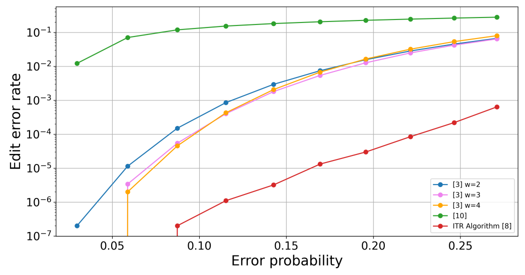


(b) Organick et al. [6].

Figure 1. Edit error rate by the reconstruction algorithm, for data from DNA storage experiments [2], [6].



(a) Success rate by the error probability.



(b) Edit error rate by the error probability.

Figure 2. Success rate and edit error rate by the error probability for t = 20. The X-axis represents the error probability of the simulated clusters and the Y-axis represents the edit error rate or the success rate, the fraction of clusters that were successfully reconstructed.

are not limited to specific cluster size, do not require any dependencies between the error probabilities, and do not assume zero errors in any specific location of the strands. Lastly, our algorithms can run with practical time on actual data from previous DNA-storage experiments. Our first algorithm, which is reffered to as the iterative reconstruction algorithm, or shortly ITR Algorithm, is inspired by the *maximum likelihood decoder* for multiple deletion channels as studied recently in [7], [9]. This algorithm is dynamic programming based and it consist of several methods that computes the *shortest common supersequence* and the *longest common subsequence* to detect common patterns that appears in the traces. The algorithm uses these patterns and their frequencies to estimate the original sequence. Our second algorithm, the divider BMA algorithm is majority based. The divider BMA algorithm runs in linear time and improves the lookahead majority technique presented in [1], [3]. Full descriptions of these algorithms, as well as more algorithms and comparison to the previously published algorithms can be found in our full paper [8].

III. Results

In this section we present an evaluation of the accuracy of our algorithms on data from previous DNA storage experiments [2], [6]. We also implemented the algorithm from [3], with lookahead window size of $w = 3$, and the algorithm from [10]. Figure 1 presents the results of the tested algorithms on data from previous DNA storage experiments [2], [6]. We clustered the data from these experiments and performed on the clusters each of the tested algorithms to evaluate their edit error rates.

Additionally, we simulated 100,000 clusters of sizes $t = 6, 10, 20$, the sequences length was $n = 100$, and the alphabet size was 4. The total error probability of deletions, insertions, and substitutions ranged between 0.029701 and 0.271. We reconstructed the original sequences of the clusters using our algorithm and the algorithm from [3] with different parameters of w, the lookahead window side. Figure 2 presents the edit error rate and the success rates, the fraction of clusters that were reconstructed successfully by the algorithms for $t = 20$.

IV. Conclusions

Even though our algorithms improved previous results, there are still several challenges that need to be addressed in order to fully solve the DNA reconstruction problem.

1) Design efficient reconstruction algorithms that improve the current edit error rate.
2) Design error correcting codes for DNA storage systems.
3) Design efficient coded trace reconstruction algorithms for DNA storage systems.
4) Standardization of DNA reconstruction algorithms.

References

[1] T. Batu, S. Kannan, S. Khanna, and A. McGregor, "Reconstructing strings from random traces," emphIn Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 910–918. Society for Industrial and Applied Mathematics, 2004.

[2] Y. Erlich and D. Zielinski, "DNA fountain enables a robust and efficient storage architecture," *Science*, vol. 355, no. 6328, pp. 950–954, 2017.

[3] P. S. Gopalan, S. Yekhanin, S. D. Ang, N. Jojic, M. Racz, K. Strauss, and L. Ceze, "Trace reconstruction from noisy polynucleotide sequencer reads," *US Patent App*, 15/536,115. 2018.

[4] R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. 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] V. I. Levenshtein, "Efficient reconstruction of sequences," *IEEE Transactions on Information Theory*, vol. 47, no. 1, pp. 2–22, 2001.

[6] L. Organick, S. D. Ang, Y.-J. Chen, R. Lopez, S. Yekhanin, K. Makarychev, M. Z. Racz, G. Ka- math, P. Gopalan, B. Nguyen, C. N. Takahashi, S. Newman, H.-Y. Parker, C. Rashtchian, K. Stewart, G. Gupta, R. Carlson, J. Mulligan, D. Carmean, G. Seelig, L. Ceze, and K. Strauss, "Random access in large-scale DNA data storage," *Nature Biotechnology*, vol. 36, pp. 242. 2018.

[7] O. Sabary, E. Yaakobi, and A. Yucovich, "The error probability of maximum-likelihood decoding over two deletion channels," *arXiv preprint*, arXiv:2001.05582, 2020.

[8] O. Sabary, A. Yucovich, G. Shapira, E. Yaakobi, "Reconstruction Algorithms for DNA-Storage Systems," *bioRxiv 2020.09.16.300186*; doi: https://doi.org/10.1101/2020.09.16.300186

[9] S. R. Srinivasavaradhan, M. Du, S. Diggavi, and C. Fragouli, "On maximum likelihood reconstruction over multiple deletion channels," *IEEE International Symposium on Information Theory (ISIT)*, pp. 436–440. 2018.

[10] K. Viswanathan and R. Swaminathan, "Improved string reconstruction over insertion-deletion channels," *The nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 399–408, 2008.