PARALLELIZING REED-SOLOMON CODES FOR ENHANCED DNA DATA STORAGE

Submitted in partial fulfilment of the requirement for the degree of Bachelor of Science Honours in Computer Science

By:
M.S.K.Alwis
2019/ASP/08

Department of Physical Science
Faculty of Applied Science
University of Vavuniya
Sri Lanka

December, 2024

1 Introduction

1.1 Background

DNA data storage leverages the unique properties of nucleotides adenine (A), thymine (T), cytosine (C), and guanine (G) to encode vast amounts of information. Each nucleotide can represent more than a bit of information, allowing for a highly dense encoding scheme [9]. This remarkable density makes DNA an attractive medium for long-term data storage solutions, with the potential to store approximately 215 petabytes (PB) of data in just one gram of DNA [5]. In comparison, traditional storage media such as hard disk drives (HDDs) and solid-state drives (SSDs) have significantly lower density, with HDDs typically storing up to 20 terabytes (TB) and SSDs ranging from 128 GB to 2 TB [1]

As a new medium, DNA offers unparalleled information density, longevity, and sustainability [5]. Its stability over time and resilience to environmental factors make it an ideal candidate for archival storage. Nevertheless, the computational processes involved in encoding and decoding DNA data with more challenges, particularly regarding error correction, which remains a significant barrier to its practical adoption [4]. RS[Reed-Solomon] codes are one of the foundational classes of error-correcting codes that serve as effective tools against noise and errors in data retrieval [2]. Despite their widespread usage, RS codes exhibit computational inefficiencies when applied to large-scale DNA storage systems [8].

While parallelization techniques have been extensively explored in other domains, such as network communication [7], their potential applications in DNA data storage remain largely untapped. This paper aims to fill this gap by investigating the parallelization of RS codes with a focus on enhancing computational efficiency without compromising error correction reliability. The study will explore how parallel computing techniques can facilitate a comparative analysis between sequential and parallelized this RS algorithm. Special emphasis will be placed on addressing DNA-specific challenges, including synthesis errors, decay, and sequencing noise [14]. The results will demonstrate scalability for RS-based solutions tailored for next-generation DNA data storage systems, laying the groundwork for computationally efficient and robust storage architectures.

Reed-Solomon codes Overview: Reed-Solomon (RS) codes are a family of error-correcting codes that were introduced by Irving S. Reed and Gustave Solomon in 1960. They are particularly effective for correcting burst errors. These algorithm mostly suitable for applications where data integrity, such as in digital communications and data storage systems. .[10]

Encoding Process: - A message represented as a vector of symbols is mapped to a polynomial p(x) of degree less than k. The codeword is generated by evaluating this polynomial at n distinct points from the finite field, resulting in a sequence of symbols that form the encoded message.

Error Correction: - The robustness of RS codes against errors comes from their ability to detect and correct multiple errors within the codeword. This capability is crucial in environments where data may be corrupted, such as during DNA synthesis and sequencing.

Application in DNA Encoding/Decoding: In the context of DNA data storage, RS codes play a role in ensuring data integrity despite the inherent challenges associated with DNA synthesis.

Keywords: DNA data storage, Reed-Solomon(RS) codes, parallel computing, error correction, computational efficiency, DNA synthesis errors, sequencing noise

1.2 Problem Definition

Despite the effectiveness of RS codes traditionally used for encoding and decoding, their application to big data can be slow [14]. This has been a limiting factor regarding the practical use of DNA data storage systems in real-time applications.

1.3 Objectives

The main goal of this research is the following:

- To parallalize RS code for encoding and decoding DNA sequences.
- To evaluate the performance improvements in terms of speed and efficiency when processing large datasets.
- To Speed up the code which make it scalable.

1.4 Motivation

The exponential growth of digital data is driving demand for immediate high-density, longterm, and sustainable storage options .[6]. To this end, DNA based information storage has evolved to one of the technologies with the greatest promise; as it exhibits superlative density, stability, and sustainability [5]. It lets vast amounts of data be stored in a compressed form, which, as such, reduces reliance on data warehouses and server farms due to the ecological impact attached to maintaining these resources. Hence, developing DNA storage further will be not only beneficiary in improving management but also one more valuable contribution to sustainability in the growing demands of technology.

However, DNA storage systems are still hindered by many challenges for practical use in real-life applications, mostly concerning efficiency in encoding and decoding of large data. RS codes, due to their robustness against errors, have been widely used for error correction in DNA storage systems [2]. This, however, comes at the cost of increased computational cost in both encoding and decoding, which could become a bottleneck for large-scale applications [8]. The elimination of this limitation will be key to making DNA data storage technologies more feasible and scalable.

A capability to break these computational bottlenecks via parallelization serves as the motivation for this study. Such advancements in parallel computing platforms are sought to be capitalized upon with OpenMP and Spark by [7]. The aim of the study is thus the achievement of faster and scalable implementations of RS codes.

2 Related Works

The field of DNA-based data storage has gained significant traction over the past decade, offering unparalleled density and durability for digital information storage. Key studies have explored encoding strategies, error correction mechanisms, and computational frameworks to optimize performance.

DNA Fountain Encoding

Erlich and Zielinski (2017) introduced the *DNA Fountain* method, a highly efficient encoding strategy for DNA data storage. While it achieved record-setting data density and robustness, computational efficiency in encoding and decoding remains an underexplored area [3]. Further advancements in combinatorial encoding, such as the DNA StairLoop approach, suggest potential directions for computational optimization in this context [14].

Error Correction with RS Codes

RS codes are widely recognized for their robust error-correcting capabilities in noisy environments. Studies such as Vardy and Bruck (1998) have developed efficient decoding algorithms for RS codes but acknowledged their computational complexity for large datasets [12]. While recent research has addressed DNA-specific challenges like decay and synthesis errors using RS codes [4], the computational bottlenecks remain significant for large-scale DNA data storage.

Parallelization Techniques for RS Codes

Parallel computing methods have demonstrated their effectiveness in accelerating RS code encoding and decoding in network communication and other domains. For example, Park and Chung (2011) explored parallel LDPC decoding using CUDA and OpenMP, while Tang and Zhang (2021) proposed an efficient parallel architecture for resource-shareable RS encoders [7, 11]. However, these studies do not address the unique constraints of DNA data storage, such as error types and synthesis limitations, nor do they evaluate performance gains from parallelization in this domain.

Emerging Error Correction Alternatives

Recent advancements, such as the *DNAe2c ECC* algorithm (2023), have improved error correction performance by modeling noise and errors specific to the DNA data channel. Similarly, soft-decision decoding strategies have demonstrated improved error correction capabilities but require further optimization for computational efficiency [2]. These methods prioritize error robustness but lack integration with parallelization techniques for RS codes.

Efficient Frameworks for DNA Storage

Frameworks designed to enhance DNA data storage efficiency, such as FrameD, address issues like DNA decay and synthesis errors and provide essential insights for optimizing storage systems [13]. Additionally, information density enhancement methods like lossy compression techniques have shown promise in reducing costs while maintaining system robustness [9]. However, these frameworks do not explicitly focus on computational efficiency for encoding and decoding processes.

Key Insights

- Parallelization techniques have proven effective for RS codes in other domains but remain underutilized for enhancing computational efficiency in DNA data storage [7, 11].
- While existing methods prioritize error correction robustness and data density, they often overlook the computational inefficiencies in RS code encoding and decoding [12, 4].
- Emerging strategies, such as soft-decision decoding and DNA StairLoop, highlight opportunities for computational optimization but require further exploration in conjunction with RS codes [2, 14].
- Frameworks like FrameD and resource-sharing architectures provide critical advancements but necessitate integration with computationally efficient encoding/decoding mechanisms [13, 11].

3 Research Gap

While RS codes are widely acknowledged as a robust error correction method for DNA-based data storage systems, their computational inefficiency particularly during encoding and decoding for large-scale datasets have significant limitation.

Recent advances have provided solutions to address DNA-specific challenges, such as soft-decision decoding to improve error correction capability [2], efficient combinatorial encoding [8], and addressing issues like DNA decay and synthesis errors [4]. Despite these contributions, several critical gaps remain:

- **Parallelization of RS Codes in DNA Data Storage:** While studies exist on parallelizing RS codes for network communication and other domains [7, 11], their direct application to DNA data storage systems has not been thoroughly investigated.
- **Computational Efficiency:** Most research does not address the computational bottlenecks inherent in RS code implementations, especially as dataset sizes scale Techniques like resource-shareable architectures [11] have optimized specific processes but fall short of addressing the end-to-end computational efficiency for encoding and decoding.
- **Application to DNA Data Storage:** Although parallel computing techniques have been implemented in other fields [7, 11], their potential for improving the efficiency of RS codes in the unique context of DNA storage systems remains largely unexplored. Some of focused for parallel processing but do not directly focus on Reed-Solomon codes [14].

3.1 Summary of the Gap

This research aims to bridge the gap by investigating the feasibility of applying parallelization techniques to Reed-Solomon codes in DNA data storage, focusing on improving computational efficiency while maintaining error correction robustness. It will provide comparative benchmarks between sequential and parallelized RS implementations, emphasizing their performance in addressing DNA-specific challenges, such as synthesis and sequencing errors. The study will also explore how parallelization can complement advancements like soft-decision decoding and resource-sharing architectures, addressing computational and scalability challenges specific to DNA-based systems.

4 Materials and Methods

4.1 Materials

- **Reed-Solomon Codes:** Find existing RS error correction code which need to enhance with parallelize techniques.
- **Computing Environment:** The implementation and testing of the algorithms will be conducted on a multi-core processor system, which allows for efficient parallel processing. The computing environment will be configured to support high-performance computing tasks, ensuring optimal resource utilization during algorithm execution.
- **Software Tools:**
  - **Programming Languages:** C++ will be employed for implementing both the single-threaded and parallelized versions of this code.

- **Parallel Computing Frameworks:** Expect to use C++ with CUDA. This choice is driven by the need to leverage the capabilities of NVIDIA GPUs to accelerate the encoding and decoding processes of the RS code.
- **Simulation Tools:** Libraries such as NumPy and Pandas will be used for data manipulation and analysis. Additionally, benchmarking tools will be integrated to assess performance metrics effectively.

4.2 Methods

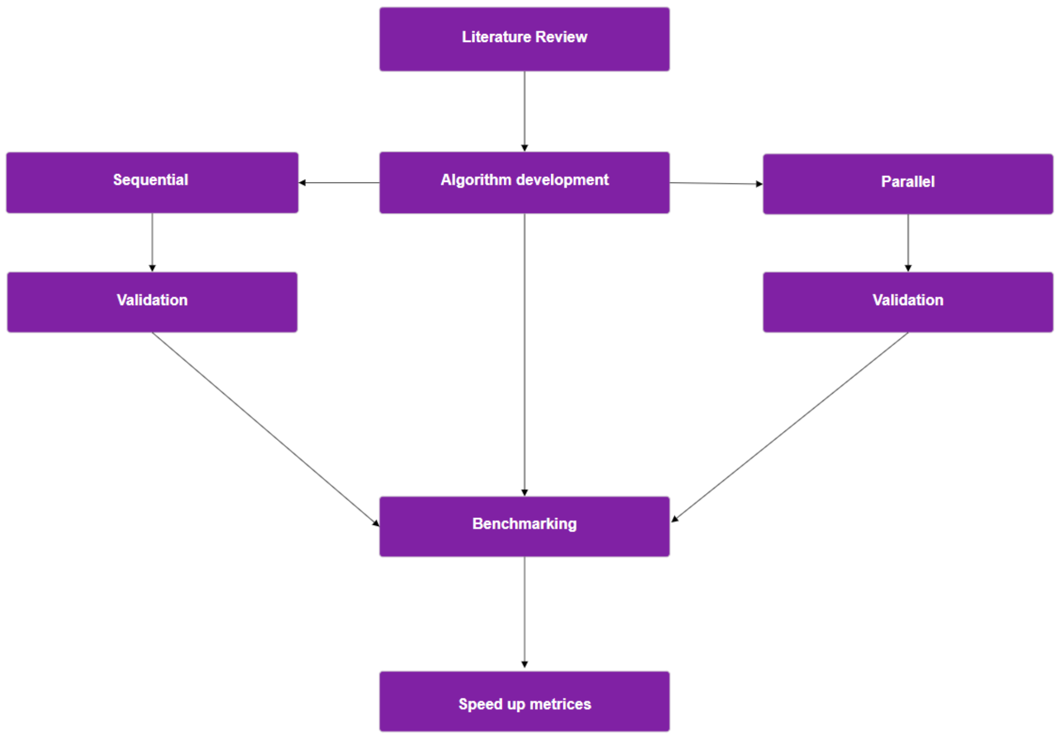

Figure 1: Methodology

1. **Literature Review:** Conduct a comprehensive review of existing studies on Reed-Solomon codes and their applications in DNA data storage. This review will identify current methodologies, limitations, and potential strategies for parallelization that have not yet been explored in the context of DNA storage.
2. **Algorithm Development:** Develop a parallelized version of the RS code, with a specific focus on optimizing both encoding and decoding processes. This development will include designing data structures that facilitate efficient memory access patterns during parallel execution.
3. **Implementation:** Implement both the single-threaded and parallelized versions of the algorithm using C++. The implementation will mostly with CUDA based on the chosen parallelization strategy to ensure scalability and performance improvements.

4. **Benchmarking:** Compare the performance of the single-threaded and parallelized algorithms based on:
    - Encoding/decoding time to evaluate speed improvements.
    - Error correction accuracy to ensure reliability in data integrity.
    - Scalability with varying dataset sizes to assess how well each implementation handles increased loads.
5. **Speed up:** Evaluate the speedup achieved by integrating the parallelized RS code into large-scale DNA data storage systems. This analysis will focus on comparing the performance improvements of the parallel implementation against the sequential code, particularly in terms of computational efficiency and processing time by measuring speedup.

5 Expected Outcome

- **Parallelized Algorithm for Reed-Solomon Codes:** Development of a parallelized version of the Reed-Solomon code for DNA data storage systems, improving the speed of encoding and decoding processes.
- **Performance Comparison:** Quantitative analysis demonstrating the computational advantages of the parallelized algorithm over traditional single-threaded implementations, including:
    - Reduced encoding and decoding time.
    - Improved scalability for larger datasets.
- **Feasibility Analysis:** Analyze whether implemented parallelize code contribute to the real world application in DNA data storage system or it leades to potentional issues which have not identified yet.

6 Ethical Approval

Not Applicable

7 Gantt Chart

Work/Time (Months)	Nov-24	Dec-24	Jan-25	Feb-25	Mar-25	Apr-25	May-25	Jun-25
Title Selection	■							
Proposal Writing and Submission		■						
Literature Review		■	■	■				
Algorithm Study and Design			■	■	■			
Tool Selection and Setup			■	■	■			
Parallel Algorithm Implementation				■	■	■		
Simulation and Testing					■	■	■	
Performance Benchmarking					■	■	■	
Writing (Thesis/Paper)					■	■	■	■
Publication / Conferences							■	■
Presentation and Submission								■

Table 1: Proposed Gantt Chart for Research Work

References

[1] A. Akash, E. Bencurova, and T. Dandekar. "How to make DNA data storage more applicable". In: *Trends in Biotechnology* (2024).

[2] Lulu Ding et al. "Improving Error-Correcting Capability in DNA Digital Storage via Soft-Decision Decoding". In: *National Science Review* 11.2 (2024). DOI: `10.1093/nsr/nwad229`.

[3] Y. Erlich and D. Zielinski. "DNA Fountain enables a robust and efficient storage architecture". In: *Science* 355.6328 (2017), pp. 950–954. DOI: `10.1126/science.aaf6846`.

[4] A.L. Gimpel et al. "Challenges for Error-Correction Coding in DNA Data Storage: Photolithographic Synthesis and DNA Decay". In: *Digital Discovery* (2024).

[5] C.P. Gomes et al. "Coding, Decoding and Retrieving a Message Using DNA: An Experience from a Brazilian Center Research on DNA Data Storage". In: *Micromachines* 15.4 (2024), p. 474. DOI: `10.3390/mi15040474`.

[6] B. Nguyen et al. "Architecting datacenters for sustainability: greener data storage using synthetic DNA". In: *Proc. Electronics Goes Green*. Vol. 105. 2020.

[7] J.Y. Park and K.S. Chung. "Parallel LDPC decoding using CUDA and OpenMP". In: *EURASIP Journal on Wireless Communications and Networking* (2011), pp. 1–8.

[8] I. Preuss et al. "Efficient DNA-based Data Storage Using Shortmer Combinatorial Encoding". In: *Scientific Reports* 14.1 (2024), p. 7731.

[9] S. Seo et al. "Information Density Enhancement Using Lossy Compression in DNA Data Storage". In: *Advanced Materials* (2024), p. 2403071. DOI: `10.1002/adma.202403071`.

[10] E. P. Slingerland. "DNA Data Storage using Hamming and Reed-Solomon Codes: DNA Data Opslag met Hamming en Reed-Solomon Codes". In: 2019. URL: `https://api.semanticscholar.org/CorpusID:198324037`.

[11] Y.J. Tang and X. Zhang. "An efficient parallel architecture for resource-shareable reed-solomon encoder". In: *2021 IEEE Workshop on Signal Processing Systems (SiPS)*. IEEE, 2021, pp. 152–157.

[12] A. Vardy and J. Bruck. "Efficient decoding algorithms for Reed-Solomon codes". In: *IEEE Transactions on Information Theory* 44.6 (1998), pp. 1984–1995.

[13] K.D. Volkel et al. "FrameD: Framework for DNA-based Data Storage Design, Verification, and Validation". In: *Bioinformatics* 39.10 (2023), btad572. DOI: `10.1093/bioinformatics/btad572`.

[14] Zihui Yan et al. "DNA StairLoop: Achieving High Error-correcting and Parallel-processing Capabilities in DNA-based Data Storage". In: *bioRxiv* (2024), pp. 2024–11.