Exploring Zstandard user-provided dictionary compression for FASTA files

Michael Persico[1]

1 Department of Biology, Concordia University, Montreal, Quebec, Canada,

☯These authors contributed equally to this work.

¤Current Address: Dept/Program/Center, Institution Name, City, State, Country

†Deceased

¶Membership list can be found in the Acknowledgments sections

Abstract

Background

Zstandard (Zstd) represents a universal, lossless data compression standard and implementation that is highly configurable and is aimed at coupling high compression ratios with fast compression/decompression performance. Previous studies have paired specific Zstd configurations with various file formats in bioinformatics to reduce total data volume. This paper presents a "training mode" pipeline, written in the Julia programming language, wherein a custom compression dictionary is generated from a sample FASTA set in order to explore further compression improvements and compare them to the compression performance of Xz, Zlib, Bzip2, and Lz4 universal compressors.

Results

Conclusions

Introduction

The explosion of biological data has represented a significant topic of research, with a number of challenges presented over subsequent generations of technological development in regards to the management of the increasing volume and complexity of data[1, 2]. In response, emerging trends in data management have lead to the development of novel, scalable methods for the efficient transmission and storage of large amounts of data[3]. With a potentially exponential quantity of files, datasets, and other data resources to be handled, data compression represents a method for reducing overall resource size by encoding the original data into a compact form, thus helping to ease storage requirements [4]. Research into data compression in the context of biological data began to pick up near the turn of the 21st century as universal compression algorithms at the time were not considered ideal for compressing DNA or RNA sequence data well, which led to the introduction of purpose-built algorithms that addressed the unique peculiarities of genomic data[5]. At the same time, new file formats were introduced, either text-based or binary-based, for more accurate structuring and representation of biological data, complementing new software tools[6, 7].

The FASTA file format is a legacy of the original FASTA program for finding sequence similarities with a query sequence[6]. Each file can possess multiple sequences, each paired with a description line distinguished by a ">" symbol followed by arbitrary text, usually a name and/or summary description, on the same line. It is a commonly supported file format in bioinformatics and has been the target for optimized data compressors with competing claims for performance. The DELIMINATE lossless algorithm was first proposed in 2012, wherein header and sequence data are separated into DELIM-1 and DELIM-2 variants and a two-phase process is pursued involving delta encoding, progressive elimination of nucleotide characters, and 7-Zip archiving[8]. The claims of better compression/decompression performance of FASTA files were soon rivaled by the introduction of the MFCompress tool, again separating headers and sequence data but instead relying on probabilistic models to encode the data[9], which was then countered by the Nucleotide Archival Format, a novel file format noteworthy in this context for the inclusion of a Zstandard compression step[10].

IETF RFC 8878, introduced by engineers at Facebook, defines Zstandard as a lossless data compression/decompression format[11]. It is often abbreviated as "Zstd", though such can also refer to Facebook's own implementation of the algorithm written mostly in C[12]. Content is sliced and packaged into "frames" that are independent of one another defined as either compressed data Zstandard frames or Skippable frames containing custom user metadata[11]. Zstd's backbone is the use of Finite State Entropy and Huffman entropy encoding schemes that replace data with coded forms independently of the medium[13, 14], with the former compressing all symbols, though header information is first encoded by the latter[11]. Zstd, for small data compression improvements, also functions as a dictionary coder, meaning that although the algorithm is universal in the sense of being applicable to a number of both text-based and binary-based data files, it can be optimized for compacting characteristic data by "training" Zstd with a collection of sample files to build a set of common patterns that allow for substitutions when compressing/decompressing, allowing for further gains for similar data[15].

Common bioinformatics file formats often include a set structure with the express purpose of representing specific kinds of biological data composed of repeating elements, as is the case with FASTA files with either nucleotide or amino acid sequences. In this work is described a pipeline for building Zstd dictionaries via FASTA datasets along with a comparison of Zstd compression/decompression performance with that of several alternative lossless compressors for select datasets.

Materials and methods

A list of all direct and indirect dependencies can be found in the repository's Manifest.toml file.

Results

Discussion

Conclusion

Supporting information

S1 Fig. Bold the title sentence. Add descriptive text after the title of the item (optional).

S2 Fig. Lorem ipsum. Maecenas convallis mauris sit amet sem ultrices gravida. Etiam eget sapien nibh. Sed ac ipsum eget enim egestas ullamcorper nec euismod ligula. Curabitur fringilla pulvinar lectus consectetur pellentesque.

S1 File. Lorem ipsum.

S1 Video. Lorem ipsum. Maecenas convallis mauris sit amet sem ultrices gravida. Etiam eget sapien nibh. Sed ac ipsum eget enim egestas ullamcorper nec euismod ligula. Curabitur fringilla pulvinar lectus consectetur pellentesque.

S1 Appendix. Lorem ipsum. Maecenas convallis mauris sit amet sem ultrices gravida. Etiam eget sapien nibh. Sed ac ipsum eget enim egestas ullamcorper nec euismod ligula. Curabitur fringilla pulvinar lectus consectetur pellentesque.

S1 Table. Lorem ipsum. Maecenas convallis mauris sit amet sem ultrices gravida. Etiam eget sapien nibh. Sed ac ipsum eget enim egestas ullamcorper nec euismod ligula. Curabitur fringilla pulvinar lectus consectetur pellentesque.

Acknowledgments

I would like to express my sincere gratitude to Professor David Walsh for his advice that helped shape the research as it progressed; My friends and family that supported me throughout my studies; The Julia community for their support before and throughout the writing of this paper; The Quarto developers for producing the Quarto publishing system[16] along with the PLOS template used for this paper; Luca Di Maio and contributors to the Distrobox tool for the ease of setting up the containerized development environments[17]; Maximiliano Sandoval and contributors to the Citations app for managing the paper's bibliography[18], and all other persons that had indirectly assisted through their programs and research.

References

1. D'Argenio V. The high-throughput analyses era: are we ready for the data struggle? High-throughput. 2018;7(1):8.
2. Li Y, Chen L. Big biological data: challenges and opportunities. Genomics, proteomics & bioinformatics. 2014;12(5):187.

3. Sais M, Rafalia N, Abouchabaka J. Intelligent Approaches to Optimizing Big Data Storage and Management: REHDFS system and DNA Storage. Procedia Computer Science. 2022;201:746–751.

4. Jayasankar U, Thirumal V, Ponnurangam D. A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications. Journal of King Saud University-Computer and Information Sciences. 2021;33(2):119–140.

5. Grumbach S, Tahi F. A new challenge for compression algorithms: genetic sequences. Information processing & management. 1994;30(6):875–886.

6. Lipman DJ, Pearson WR. Rapid and sensitive protein similarity searches. Science. 1985;227(4693):1435–1441.

7. Mills L. Common file formats. Current protocols in bioinformatics. 2014;45(1):A–1B.

8. Mohammed MH, Dutta A, Bose T, Chadaram S, Mande SS. DELIMINATE—a fast and efficient method for loss-less compression of genomic sequences: Sequence analysis. Bioinformatics. 2012;28(19):2527–2529.

9. Pinho AJ, Pratas D. MFCompress: a compression tool for FASTA and multi-FASTA data. Bioinformatics. 2014;30(1):117–118.

10. Kryukov K, Ueda MT, Nakagawa S, Imanishi T. Nucleotide Archival Format (NAF) enables efficient lossless reference-free compression of DNA sequences. Bioinformatics. 2019;35(19):3826–3828.

11. Collet Y, Kucherawy M. RFC8478: Zstandard compression and the application/zstd media type; 2021.

12. Facebook. Facebook/ZSTD: Zstandard - fast real-time compression algorithm;. Available from: https://github.com/facebook/zstd.

13. Ezhilarasan M, Thambidurai P, Praveena K, Srinivasan S, Sumathi N. A new entropy encoding technique for multimedia data compression. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007). 2007;doi:10.1109/iccima.2007.123.

14. Lu ZM, Guo SZ. Chapter 1 - Introduction. In: Lu ZM, Guo SZ, editors. Lossless Information Hiding in Images. Syngress; 2017. p. 1–68. Available from: https://www.sciencedirect.com/science/article/pii/B9780128120064000012.

15. Kanda S, Morita K, Fuketa M. Practical String Dictionary Compression Using String Dictionary Encoding. In: 2017 International Conference on Big Data Innovations and Applications (Innovate-Data); 2017. p. 1–8.

16. Allaire JJ, Teague C, Scheidegger C, Xie Y, Dervieux C. Quarto; 2022. Available from: https://github.com/quarto-dev/quarto-cli.

17. Di Maio L. Distrobox;. Available from: https://github.com/89luca89/distrobox.

18. Sandoval M. Citations – apps for Gnome;. Available from: https://apps.gnome.org/app/org.gnome.World.Citations/.