The K-mer File Format: a standardized and compact disk representation of sets of k-mers
Yoann Dufresne, Teo Lemane, Pierre Marijon, Pierre Peterlongo, Amatur Rahman, Marek Kokot, Paul Medvedev, Sebastian Deorowicz, Rayan Chikhi

To cite this version:
Yoann Dufresne, Teo Lemane, Pierre Marijon, Pierre Peterlongo, Amatur Rahman, et al.. The K-mer File Format: a standardized and compact disk representation of sets of k-mers. Bioinformatics, 2022, 38 (18), pp.4423-4425. 10.1093/bioinformatics/btac528. hal-03885245

HAL Id: hal-03885245
https://inria.hal.science/hal-03885245
Submitted on 9 Dec 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
Sequence analysis

The K-mer File Format: a standardized and compact disk representation of sets of k-mers

Yoann Dufresne, Teo Lemane, Pierre Marijon, Pierre Peterlongo, Amatur Rahman, Marek Kokot, Paul Medvedev, and Rayan Chikhi

1Computational Biology Department, Institut Pasteur, Université Paris Cité, F-75015 Paris, France, 2Univ Rennes, Inria, CNRS, IRISA—UMR, 6074 Rennes, France, 3Heinrich Heine University Düsseldorf Medical Faculty Institute for Medical Biometry and Bioinformatic, Düsseldorf 40225, Germany, 4Department of Computer Science and Engineering, The Pennsylvania State University, State College 16802, USA, 5Department of Algorithmics and Software, Silesian University of Technology, Gliwice, PL-44-100 Akademicka 16, Poland, 6Department of Biochemistry and Molecular Biology, The Pennsylvania State University, State College 16801, USA and 7Huck Institutes of the Life Sciences, The Pennsylvania State University, State College 16802, USA

*To whom correspondence should be addressed.

Abstract

Summary: Bioinformatics applications increasingly rely on ad hoc disk storage of k-mer sets, e.g. for de Bruijn graphs or alignment indexes. Here, we introduce the K-mer File Format as a general lossless framework for storing and manipulating k-mer sets, realizing space savings of 3-5x compared to other formats, and bringing interoperability across tools.

Availability and implementation: Format specification, C++/Rust API, tools: https://github.com/Kmer-File-Format/.

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

Sets of k-mers are widely used in DNA sequence analysis, for instance in genome assembly [e.g. SPAdes (Bankevich et al., 2012)], indexes of sequence aligners [e.g. minimap2 (Li, 2018)], large-scale sequence search tools [Marchet et al., 2021]. Often, bioinformatics tools are k-mer consumers, i.e. they take as input a k-mer set given by one of the k-mer producers, typically k-mer counters [e.g. KMC (Deorowicz et al., 2013), DSK (Rizk et al., 2013)]. Producers use ad hoc binary formats for storing k-mers on disk. This leads to inefficient development practices, as consumers need to write specific parsers for each producer format. Standard file formats greatly facilitate interoperability, e.g. in the case of the SAM/BAM formats (Cock et al., 2015) for sequence alignment and HDF5 (Folk et al., 2011) for general structured data.

We propose the K-mer File Format (KFF), an interoperable and efficient approach to store k-mers. We provide APIs in C++ and Rust, as well as file manipulation and conversion tools to facilitate inspection and integration into other tools. KFF has already been integrated in several tools: the KMC and DSK k-mer counters, the ESS-Compress (Rahman et al., 2020) compression tool and kmtricks (Lemane et al., 2022) for k-mer matrix construction. We present the rationale of our approach, the KFF 1.0 file format, and demonstrate the efficiency of KFF for storing k-mers from sequencing data.

2 Approach

Tools producing k-mer sets essentially use similar storage techniques. In Jellyfish (Marcáis and Kingsford, 2011) and DSK, a k-mer is encoded in 2 bits per nucleotide and the entire set is stored as a succession of k-mers and associated data (e.g. abundances). In KMC, a more advanced format is used to reduce space and to allow fast, logarithmic time, queries (see ‘KMC file format description’ in the Supplementary Material for more details).

Recent works (Brinda et al., 2021; Rahman et al., 2021) demonstrated space-efficient storage of genomic k-mers using their spectrum-like property (Chikhi et al., 2021), i.e. assuming that most k-mers originate from a set of long strings. In this spectrum-preserving string set representation (SPSS), what are stored are sequences longer than k, where each window of length k is a k-mer from the original set, and achieve a space of around 3 bits per k-mer [Rahman et al. (2020), k = 31, no counts stored]. However, the representation is non-trivial to compute and requires hours for a human genome.

We propose a space-efficient format that is fast to produce, encoding k-mers in binary and storing them in overlapping form. The drawback for space efficiency is that random accesses are not supported in KFF, yet they are unnecessary in the many consumer applications that only read k-mer sets from disk sequentially (Bankevich et al., 2012; Rahman et al., 2020).
3 Methods

A KFF file is composed of a short header and a succession of sections (see Fig. 1). The header contains the format version, the nucleotide 2-bit encoding (e.g. A:0, C:1, G:3, T:2), global flags to indicate sequences (which are helpful for the following sections. A V section indicates its type. A V section defines variables that are helpful for the following sections. Actual storage of sequences is done in sections which contain the same set of k-mers. For example, the first entry in the T section has sequence ACTG with its minimizer at position 3, hence it corresponds to sequence ACTGATCGAGT (which is identical to the first entry in the R section), from which three k-mers can be extracted

4 Results

We created the kff-tools software suite on top of the C++ KFF API. It is a collection of small programs that manipulate KFF files, such as merging/splitting, validation, bucketing. They are available at github.com/Kmer-File-Format/kff-tools. These tools complement the already existing KMC tools (Kokot et al., 2017) that allow more complex operations on k-mer sets, e.g. union, intersection and comparison pending). On average KFF with super-k-mers requires 17 bits per k-mer (omitting the data), while KMC uses 56 bits/k-mer. Using SPSS improves storage further to 5 bits per k-mer. Furthermore, gap compression adds an additional 2× compression gain for KFF files and 1.25× gain for KMC files.

In conclusion, we propose the k-mer set file format KFF, along with a versatile C++ and Rust API to read and write k-mers and a toolkit for file manipulations. We hope that KFF will boost interoperability between many software tools that use k-mer sets, and simultaneously improve their efficiency due to the compression features of KFF. Many suggestions and requests are emerging from discussions with the community and extensions of features to the format are currently being considered. The KFF format could for instance be used to store super-k-mers sketches, although current sketching tools store hashes on disk (Pierce et al., 2019), discarding the originating k-mers.

Acknowledgements

The authors thank Dr Maggie Sefton for help with Figures.

Funding

R.C. was funded by ANR Inception (ANR-16-CONV-0005) and PRAIRIE (ANR-19-P3IA-0001) grants. M.K. and S.D. were supported by the National Science Centre, Poland [DEC-2019/33/B/ST6/02040]. Work supported in part by the National Science Foundation [1453527 and 1931531], and by the European Union’s Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie grant agreement No 956229.

Conflict of Interest: The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Bankevich,A. et al. (2012) SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. J. Comput. Biol., 19, 455–477.

Brinda,K. et al. (2021) Simplings as an efficient and scalable representation of de Bruijn graphs. Genome Biol., 22, 1–24.

Chikhi,R. et al. (2021) Data structures to represent a set of k-long DNA sequences. ACM Comput. Surv., 54, 1–22.

Table 1. Comparison of file sizes (in GB) for several techniques for storing 32-mers on disk: naive plain-text encoding (‘T’), KMC file format (‘KMC’), KFF file format storing one k-mer per block (‘KFF-naive’) or storing super-k-mers as created by kmtricks (‘KFF+sk’), or using k-mers stored as a string-preserving string set (‘KFF+SPSS’).

| Sample     | T       | KMC     | KFF+naive | KFF+sk | KFF+SPSS |
|------------|---------|---------|-----------|--------|----------|
| Gallus gallus | 95.1    | 19.1    | 24.2      | 7.4    | 4.2      |
| G.gallus, gz | 19.9    | 15.0    | 16.6      | 4.8    | 2.0      |
| Homo sapiens | 191.0   | 37.7    | 48.5      | 16.8   | 11.1     |
| H.sapiens, gz | 37.9    | 30.6    | 33.8      | 11.9   | 6.4      |

Note: ‘gz’ indicates gzip compressed outputs.
Cock, P.J., et al. (2015) Sam/bam format v1.5 extensions for de novo assemblies. *BioRxiv*, page 020024. https://doi.org/10.1101/020024.

Deorowicz, S. et al. (2013) Disk-based k-mer counting on a PC. *BMC Bioinformatics*, 14, 160.

Folk, M. et al. (2011) An overview of the hdf5 technology suite and its applications. In: *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, ACM, Uppsala, Sweden, pp. 36–47.

Kokot, M. et al. (2017) KMC 3: counting and manipulating k-mer statistics. *Bioinformatics*, 33, 2759–2761.

Lemane, T. et al. (2022) kmtricks: Efficient construction of bloom filters for large sequencing data collections. *Bioinformatics Adv.*, 2. https://doi.org/10.1093/bioadv/vbac029.

Li, H. (2018) Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34, 3094–3100.

Marçais, G. and Kingsford, C. (2011) A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, 27, 764–770.

Marchet, C. et al. (2021) Data structures based on k-mers for querying large collections of sequencing data sets. *Genome Res.*, 31, 1–12.

Pierce, N.T. et al. (2019) Large-scale sequence comparisons with sourmash. *F1000Res.*, 8, 1006.

Rahman, A. et al. (2021) Representation of k-mer sets using Spectrum-Preserving string sets. *J. Comput. Biol.*, 28, 381–394.

Rahman, A. et al. (2020) Disk compression of k-mer sets. In: 20th International Workshop on Algorithms in Bioinformatics. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, Pisa, Italy.

Rizk, G. et al. (2013) DSK: k-mer counting with very low memory usage. *Bioinformatics*, 29, 652–653.