Genie: an MPEG-G conformant software to compress genomic data

B Bliss[1], J. Allen[1], S. Baheti[3], M. A. Bockol[3], S. Chandak[2], J. Delgado[4], J. Fostier[5], J.L. Gelpi[8], S. N. Hart[3], M. Hernaez Arrazola[1], M. E. Hudson[1], M. T. Kalmbach[3], E. W. Klee[3], L. S. Mainzer[1], F. Müntefering[6], D. Naro[7], I. Ochoa-Alvarez[1], J. Ostermann[6], T. Paridaens[5], C. A. Ross[3], J. Voges[6], E. D. Wieben[3], M. Yang[1], T. Weissman[2], M. Wiepert[3]

Statement of the problem

Precision medicine has been identified as a national priority [1,2] and world wide concern [3-7], because it has greater potential for accurate diagnosis than traditional medicine, and ability to tailor treatment to the patient, resulting in lower cost of care and faster recovery, e.g., in cancer treatment, autoimmune disorders and dementia. This led to an explosion of genomic data, which will continue to accumulate at a rate that rivals or exceeds that in astronomy and social media [8]. Yet storage and analysis of Petascale genomic data is very expensive, and that cost will ultimately be borne by the patients and citizens. There exists an urgent need to evolve from the current file formats FASTQ and BAM/SAM to a data representation that facilitates efficient compression [9], selective access [10], transport and analysis [9]. Several critical barriers hinder the adoption of efficient specialized formats: i) poor guarantee for long-term support; ii) technical limitations for selective access on the compressed data; and iii) poor support for integrated annotation and encryption of compressed genomic information.

MPEG-G as a solution

- Moving Picture Experts Group (MPEG) is a joint working group of the International Standardization Organization (ISO) and the International Electrotechnical Commission (IEC)
- MPEG has developed a **new open standard** [11] to compress, store, transmit and process genomic sequencing data, called MPEG-G (https://mpeg.chiariglione.org/standards/mpeg-g).
- A detailed specification was generated, embedding mechanisms to **resolve the above technical difficulties** in securing, storing, and moving petascale genomic data.
- ISO backing provides the assurance of long-term support.

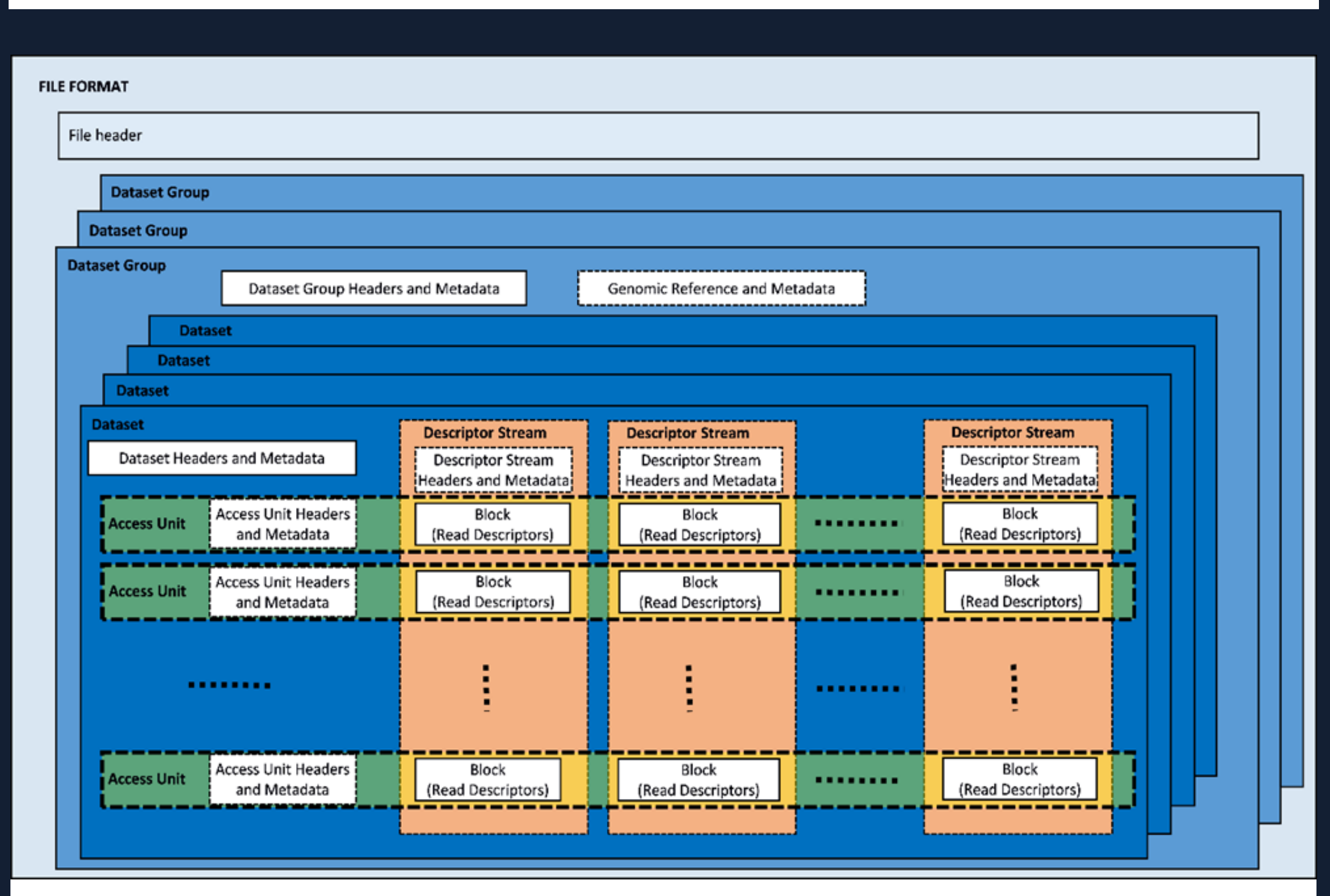
FILE FORMAT
File header
Dataset Group
Dataset Group
Dataset Group
Dataset Group Headers and Metadata
Genomic Reference and Metadata
Dataset
Dataset
Dataset
Dataset
Dataset Headers and Metadata
Descriptor Stream
Descriptor Stream Headers and Metadata
Access Unit
Access Unit Headers and Metadata
Block (Read Descriptors)

Figure 1: MPEG-G file example on sequencing data of a trio. The MPEG-G file can encapsulate the entire genomic history of one or more individuals in a unique file including the metadata describing the study, samples, etc. For example, in a trio:

- File Header: metadata related to the study.
- Dataset Group: one per individual + metadata from the individual.
- Dataset: sequencing data + metadata from one experiment.
- Colored boxes: how genomic data are represented in MPEG-G.

GENIE as a software implementation

We have developed GENIE (Figure 2), the first open source implementation of an encoder-decoder pair that is compliant with the MPEG-G specifications and delivers all its benefits. GENIE is now focused on compression, but also supports development of efficient data transfer and APIs for operating directly on the compressed data. It supports lossless and lossy compression of genomic data in the form of FASTA, FASTQ and SAM files and is based on the FAIR (Findable, Accessible, Interoperable, and Reusable) principles. https://github.com/mitogen/genie/.

GENIE software architecture

The unaligned input data are split into streams (Figure 3) to separate out read IDs and sequences from the quality scores. These streams are directed into SPRING and CALQ, respectively, for initial processing and conversion into *descriptor streams* that are maximally compressible by GABAC. GABAC (Figure 4) is our rendition of the popular CABAC (entropy encoding for video) that is specifically designed for genomic sequences. SPRING and CALQ are also tools developed by our team. They are tied together as library modules within GENIE, which packages the compressed data into a single output file in a format that follows the MPEG-G specification.

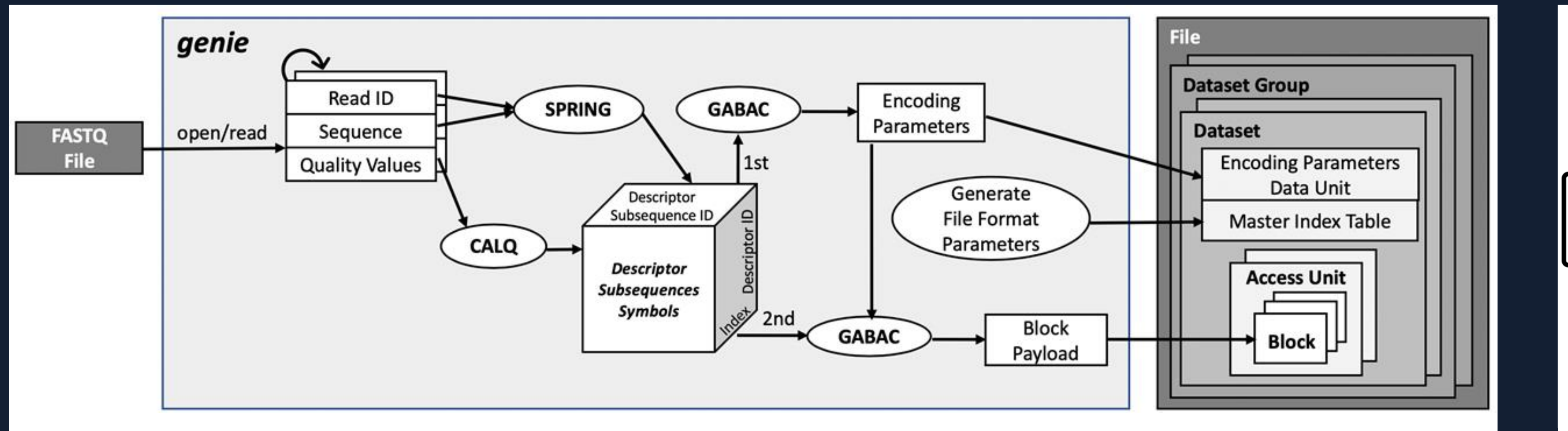

Figure 2: Internal organization of GENIE. The light gray shaded box on the left circumscribes the GENIE repertoire of data stream generation, compression and encoding. The gray shaded box on the right displays the MPEG-G conformant structure of the output data file format.

GABAC: the compression core of GENIE

Given an input stream, the compression process consists of a five stage pipeline: (1) input parsing, (2) (optional) 3-step transformation, (3) symbol binarization, (4) context selection, and (5) CABAC. First the input descriptor stream is parsed into a stream of symbols. These symbols are processed by the 3-step transformation stage that converts the symbol stream into transformed sub-streams. For each transformed sub-stream, a binarization algorithm converts each symbol into a bit string. It is chosen together with a context selection algorithm. Finally, each bit of the binarization is combined with a context and both are processed using CABAC.

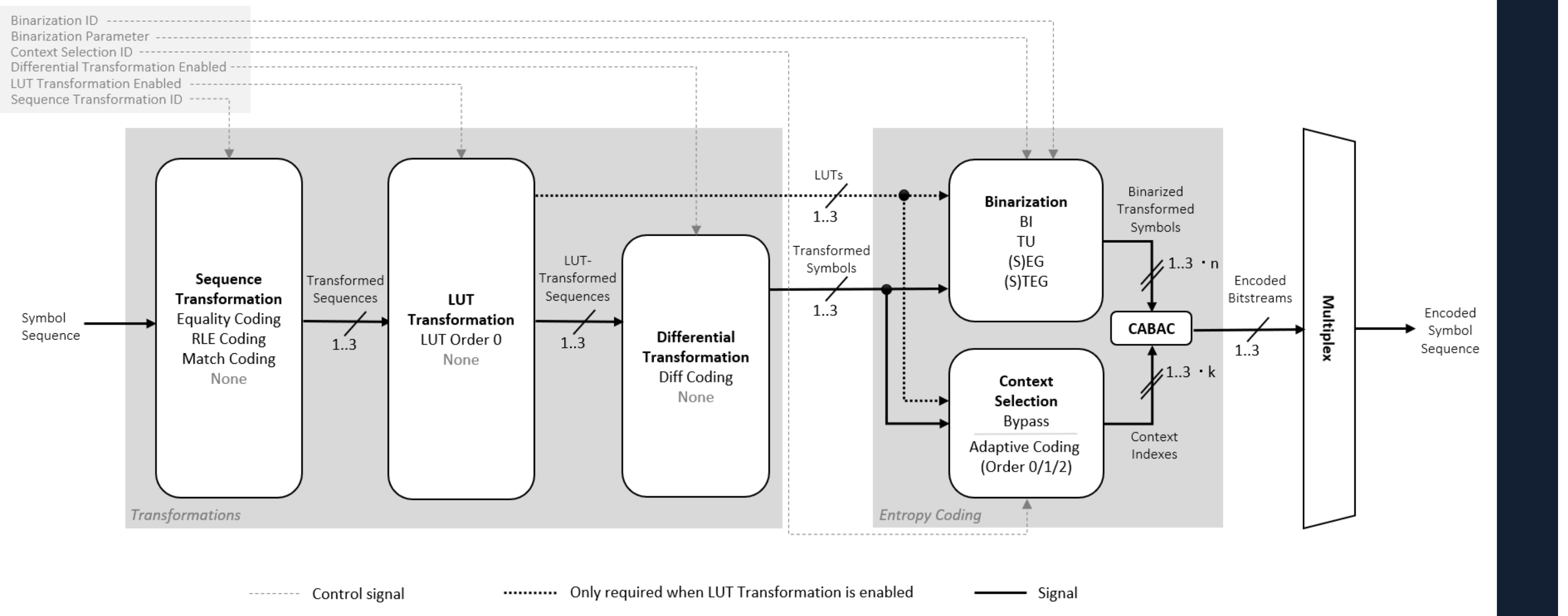

Figure 4: Internal organization of GABAC. Among the 5 stages of the GABAC pipeline, data transformation is very important for maximal compression of genomic data. The transformation steps and possible kinds of transformation are explained inside the left gray shaded rectangle on the figure.

GENIE internal data stream generation

- Each of the salient data types within a FASTQ file, such as the read names, sequences and quality scores, have very distinct entropic properties, and therefore cannot all be maximally compressed using the same entropy model.
- We split the FASTQ data into streams, one for each data types, and prepare them for compression with the entropy coding model that is most appropriate for that data type.

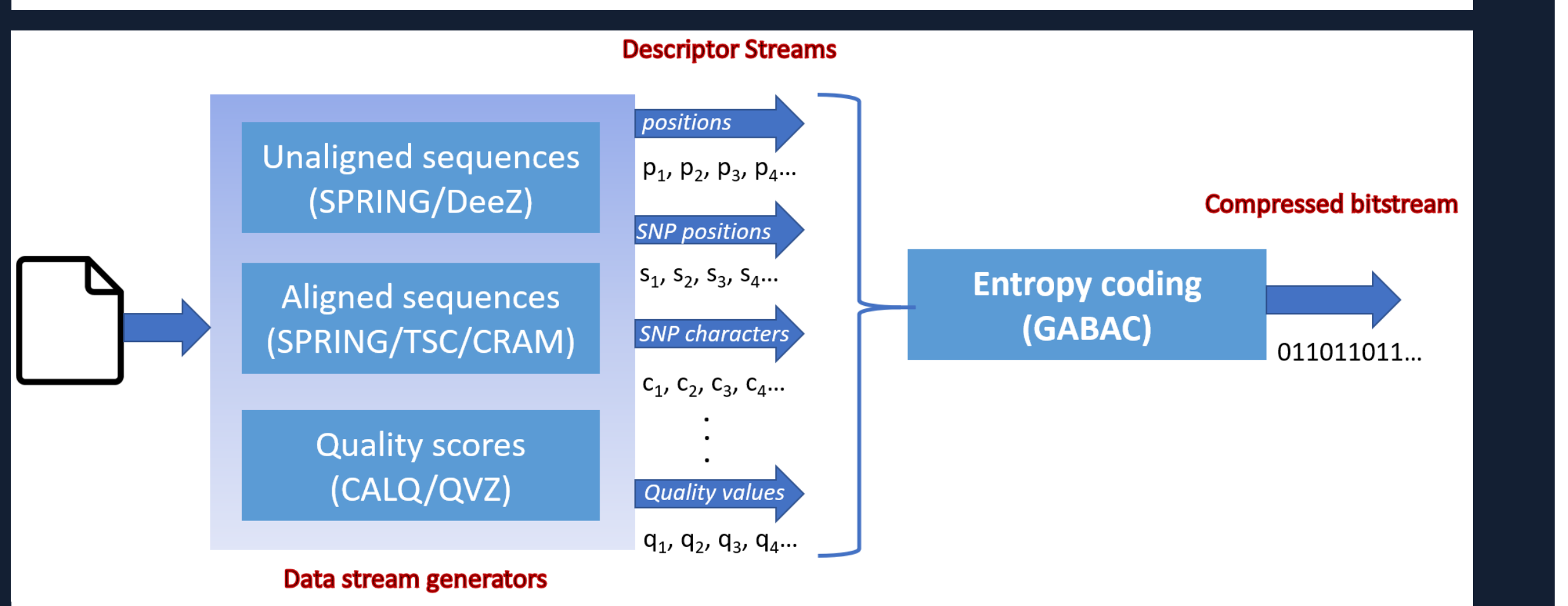

Figure 3: Data are split into streams for best compression ratio. The GABAC module has provisions for specifying correct kind of transformation for each data type, in preparation for maximal compression.

Community and Project governance

- Our team is a diverse international developer community (see list of organizations under Acknowledgements).
- Together we have developed several methods to reduce the footprint of genomic information, ranging from specialized compressors for raw sequencing data to novel ways of representing genomic variants that enable efficient selective access on large populations.
- We are also active participants in the MPEG-G working group and have made several technical contributions to it.
- As a result of this multipronged effort, we developed infrastructure to support our collaboration via:
  - a code of conduct,
  - an issue-tracker,
  - contribution and coding guidelines, and
  - continuous integration on the code repository:

https://github.com/mitogen/genie/blob/master/CODEOFCONDUCT.md
https://github.com/mitogen/genie/issues
https://github.com/mitogen/genie/blob/master/PULLREQUESTTEMPLATE.md
https://travis-ci.org/mitogen/genie

The Pirbright Institute — Stanford University — ILLINOIS University of Illinois at Urbana-Champaign — Ghent University — UPC Universitat Politècnica de Catalunya BarcelonaTech — Leibniz Universität Hannover

Conclusions and next steps

- GENIE is a **novel** implementation of genomic data compression that is compliant with the new MPEG-G encoding standard
- It will reduce the cost of genomic data storage by an order of magnitude, **enabling petascale analyses at fraction of cost**.
- It is a package constructed from several opensource codes that we integrated into a single application.
- Parallelization is implemented with OpenMP across all modules.
- We are working to replace intermediary files as the mechanism of communication between SPRING and GABAC with streaming in RAM, which **will lighten the load on the filesystem**.
- Next we will optimize the OpenMP constructs to extract the maximum benefit from the available parallelism and prevent any thread contention issues.
- The workload uniformity and overhead involved in forking parallel regions will be profiled to ensure good scalability.
- Code is being hardened for long-term maintenance and robust production use via extensive unit tests, inline documentation, internal error checks, logging and meaningful error messages.

Acknowledgements

This work was a product of the Mayo Clinic and Illinois Strategic Alliance for Technology-Based Healthcare. Major funding was provided by the Mayo Clinic Center for Individualized Medicine and the Todd and Karen Wanek Program for Hypoplastic Left Heart Syndrome. We thank the Interdisciplinary Health Sciences Institute, UIUC Institute for Genomic Biology and the National Center for Supercomputing Applications for their generous support and access to resources. We particularly acknowledge the support of Keith Stewart, M.B., Ch.B., Mayo Clinic/Illinois Grand Challenge Sponsor and Director of the Mayo Clinic Center for Individualized Medicine. Special gratitude to Amy Weckle, Katherine Kendig and Gay Reed for managing the project. We are also grateful for the support of the Chan Zuckerberg Initiative DAF (2018-182798 and 2018-182799).

ORGANIZATIONS
1. University of Illinois at Urbana-Champaign, Urbana, IL, USA
2. Stanford University, Stanford, CA, USA
3. Mayo Clinic, Rochester, Minnesota, USA
4. Universitat Politecnica de Catalunya, Barcelona, Spain
5. IDLab, Ghent University, Ghent, Belgium
6. Institut für Informationsverarbeitung (TNT), Leibniz University Hannover, Germany
7. Barcelona Supercomputing Center, Barcelona, Spain
8. Universitat de Barcelona, Barcelona, Spain

REFERENCES
1. https://obamawhitehouse.archives.gov/precision-medicine
2. https://www.asco.org/advocacy-policy/asco-in-action/national-institutes-health-launches-new-precision-medicine-initiative
3. Garrido P, Aldaz A, Vera R, Calleja M, de Alava E, Martín M, et al. Proposal for the creation of a national strategyfor precision medicine in cancer: a position statement of SEOM, SEAP, and SEFH. Clinical and TranslationalOncology. 2018;20(4):443–447.
4. Marx V. The DNA of a nation. Nature Publishing Group; 2015.
5. Njølstad PR, Andreassen OA, Brunak S, Børglum AD, Dillner J, Esko T, et al. Roadmap for a precision-medicineinitiative in the Nordic region. Nature genetics. 2019;51(6):924.
6. Lu JfR, Eggleston K, Chang JTC. Economic Dimensions of Personalized and Precision Medicine in Asia. EconomicDimensions of Personalized and Precision Medicine. 2019;p. 237.
7. Chung B, Willis B, Lai PS. Development of clinical genetics in Asia. In: American Journal of Medical Genetics PartC: Seminars in Medical Genetics. vol. 181. Wiley Online Library; 2019. p. 150–154
8. Stephens ZD, Lee SY, Faghri F, Campbell RH, Zhai C, Efron MJ, et al. Big data: astronomical or genomical? PLoSbiology. 2015;13(7):e1002195
9. Greenfield D, Wittorff V, Hultner M. The Importance of Data Compression in the Field of Genomics. IEEE pulse.2019;10(2):20–23.[12]
10. Banerjee SS, Athreya AP, Mainzer LS, Jongeneel CV, Hwu WM, Kalbarczyk ZT, et al. Efficient and scalableworkflows for genomic analyses. In: Proceedings of the ACM International Workshop on Data-Intensive DistributedComputing. ACM; 2016. p. 27–36.
11. Alberti C, Paridaens T, Voges J, Naro D, Ahmad JJ, Ravasi M, et al. An introduction to MPEG-G, the new ISOstandard for genomic information representation. bioRxiv. 2018;p. 426353.

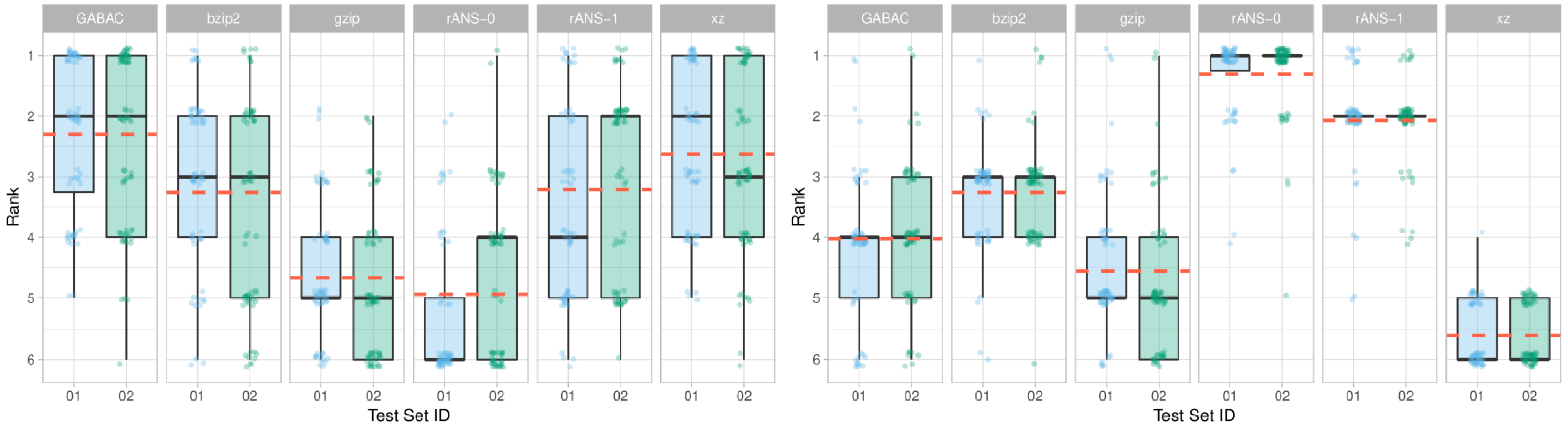

Figure 5: Compression ratios (left) and speed (right) across several codecs. GABAC yields the best compression ratios and is faster than gzip and xz in its optimal configuration.

Each codec was run on human whole genome sequencing chromosome 11 data (items 01 and 02 in https://mpeg-g.org, BAM files are 6.9 GB and 4.2 GB in size, respectively). To make codecs comparable, we modified the compression tools CRAM and DeeZ to enable access to their internal data representations. These data were used as descriptor streams, each encoded with the entropy codecs used in CRAM (gzip, bzip2, xz, rANS order-0 or rANS order-1), plus GABAC. To further emulate block-wise compression (random access capabilities), all streams were limited to 200 MiB. This approach allows for a more extensive test set in a random access environment, while preserving a reliable representation of the coding performance for each of the compared codecs.

Measurements of compression ratio and speed on each descriptor stream were **ranked**, and the **rank plotted on the vertical axis** for the different input datasets. The dotted red lines denote the mean rank for each codec, averaged over both test items. As a proxy for the spread between the ranks we computed the average compression ratios and speeds for the codecs that rank first (22% and 49 MiB/s, respectively) and the set of codecs that rank last (34% and 2 MiB/s, respectively).