Executive Summary

ADAM: Genomics Formats and Processing Patterns for Cloud Scale Computing

Matt Massie[1], Frank Austin Nothaft[1], Chris Hartl[1,2], Christos Kozanitis[1], André Schumacher[3], Anthony D. Joseph[1], and David Patterson[1]

[1]Department of Electrical Engineering and Computer Science, University of California, Berkeley
[2]The Broad Institute of MIT and Harvard
[3]International Computer Science Institute (ICSI), University of California, Berkeley

1 Introduction

The overall volume of genomic data is rapidly increasing, thanks to a significant drop in sequencing costs [3]. Current genomics data formats and processing pipelines are not designed to scale well to large datasets. The current Sequence/Binary Alignment/Map (SAM/BAM) formats were intended for single node processing [2]— there have been attempts to adapt BAM to distributed computing environments, but they see limited scalability past 8 nodes [4]. Additionally, due to the lack of an explicit data schema, there are well known incompatibilities between libraries that implement SAM/BAM/Variant Call Format (VCF) data access.

To address these problems, we introduce ADAM, a set of formats and processing stage implementations for genomic data. ADAM is fully open source under the Apache 2 license, and is implemented on top of Avro and Parquet [1, 5] for data storage. Our reference pipeline is implemented on top of Spark, a high performance in-memory map-reduce system [6]. This provides the following advantages:

- Avro provides explicit data schema access in C/C++/C#, Java/Scala, Python, php, and Ruby
- Parquet allows access by database systems like Impala and Shark
- Spark improves performance through in-memory caching and reduces I/O

In addition to improving the format's cross-platform portability, these changes lead to significant performance improvements. On a single node, we are able to improve sort and duplicate marking performance by 2×. On a 250 Gigabyte (GB) high (60×) coverage human genome, this system achieves a 50× speedup on a 100 node computing cluster.

2 ADAM Data Format

The ADAM format provides explicit schemas for read and reference oriented (pileup) sequence data, variants, and genotypes. As explained in the introduction, the schemas are implemented in Apache Avro, which is a cross-platform/language serialization format. Avro eliminates the need for the development of language-specific libraries for format decoding/encoding, which eliminates the possibility of library incompatibilities.

A key feature of ADAM is that any application that implements the ADAM schema is compatible with ADAM. This is important, as it prevents applications from being locked into a specific tool or pattern. We envision that ADAM enables the stack shown in Figure 1. This stack is inspired by the "narrow waist" of the Internet Protocol (IP) suite.

In addition to the advantages outlined above, ADAM eliminates the file headers in modern genomics formats. All header information is available inside of each individual record. The variant and genotype formats also demonstrate two significant improvements. First, these formats are co-designed so that variant data can be seamlessly calculated from a given collection of sample genotypes. Secondly, these formats are designed to flexibly accommodate annotations without cluttering the core variant/genotype schema. In addition to the benefits described above, ADAM files are 25% smaller on disk than compressed BAM files.

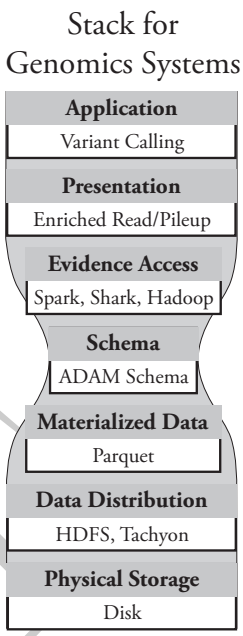

Figure 1: Stack Model

3 Pipeline Construction and Performance

The ADAM processing pipeline uses Spark as a compute engine and Parquet for data access. Spark is an in-memory MapReduce framework which minimizes I/O accesses. We chose Parquet for data storage as it is an open-source columnar store that is designed for distribution across multiple computers with high compression. Additionally, Parquet supports efficient methods (predicates and projections) for accessing only a specific segment or fields of a file, which can provide significant (2-10×) speedups for genomics data access patterns.

At this date, we have implemented Sort and Duplicate Marking in the ADAM pipeline. BQSR and Indel Realignment implementations are forthcoming. The performance of these stages is shown in Table 1.

Table 1: Sort (L) and Duplicate Marking (R) for NA12878

Software	EC2 profile	Wall Clock Time
Picard 1.103	1 `hs1.8xlarge`	17h 44m
ADAM 0.5.0	1 `hs1.8xlarge`	8h 56m
ADAM 0.5.0	32 `cr1.8xlarge`	33m
ADAM 0.5.0	100 `m2.4xlarge`	21m

Software	EC2 profile	Wall Clock Time
Picard 1.103	1 `hs1.8xlarge`	20h 22m
ADAM 0.5.0	100 `m2.4xlarge`	29m

These tests were run on a high coverage (60×) full genome sequencing of NA12878.

4 Conclusion

This executive summary and the attached technical report introduce ADAM, a data format that is optimized for high performance distributed genomic data processing pipelines. ADAM is a modular system that is open source under the Apache 2 license. On traditional workloads, ADAM provides a >50× speedup, and ADAM files are 25% smaller than their compressed BAM equivalents.

References

[1] Apache. Avro. `http://avro.apache.org`.

[2] Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., et al. The sequence alignment/map format and SAMtools. *Bioinformatics 25*, 16 (2009), 2078–2079.

[3] NHGRI. DNA sequencing costs. `http://www.genome.gov/sequencingcosts/`.

[4] Niemenmaa, M., Kallio, A., Schumacher, A., Klemelä, P., Korpelainen, E., and Heljanko, K. Hadoop-BAM: directly manipulating next generation sequencing data in the cloud. *Bioinformatics 28*, 6 (2012), 876–877.

[5] Twitter, and Cloudera. Parquet. `http://www.parquet.io`.

[6] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., and Stoica, I. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (2010), p. 10.