ADAM Enables Large Scale Integrative Analyses Across Heterogenous Genomic Datasets

Frank Austin Nothaft[1], et al[1,2,3,4,5,6]

[1] *AMPLab, University of California, Berkeley, CA*

[2] *Cloudera, Inc., San Francisco, CA*

[3] *GenomeBridge, Cambridge, MA*

[4] *Icahn School of Medicine at Mount Sinai, New York, NY*

[5] *Genome Informatics Lab, University of Californa, Santa Cruz, CA*

[6] *Microsoft Research, Redmond, WA*

The detection and analysis of rare genomic events requires integrative analysis across large cohorts with terabytes to petabytes of genomic data. Contemporary genomic analysis tools have not been designed for this scale of data-intensive computing. In this paper, we present ADAM, a library built on top of the popular Apache Spark distributed computing framework. ADAM provides high level primitives that enable genomic analyses to be efficiently executed across a large cluster of computers. Unlike other toolkits, ADAM is designed to easily enable the mixing of different types and sources of genomic data. In this paper, we demonstrate ADAM using several applications that process large genomic feature, genotype, and read datasets. ADAM enables analyses to be seamlessly distributed across hundreds of nodes, while achieving a 66% cost improvement over current toolkits.

1 Introduction

1. A big driver for extensive sequencing is detecting low frequency genomic correlations

   (a) This necessitates the joint analysis/integration of large cohorts; these datasets are frequently too large to process on a single machine

   (b) The GATK provides common primitives for constructing genomic analyses [1], but is difficult to run in a distributed setting

   (c) We want to leverage the past decade of industrial development of horizontally scalable systems [2, 3]

   (d) This approach has been successfully applied to neuroscience [4]

2. In ADAM, we provide abstractions for accessing and processing genomic data on a cluster of commodity machines

   (a) We provide a programing framework that accelerates common genomic processing tasks, while not limiting the algorithms that can be run

   (b) Algorithms are high level: shouldn't be concerned about data formats or the layout of data on disk

   (c) Our approach enables the integration of many different types of genomic datasets, which has been overlooked by current platforms [5]

3. By taking a "ground up" approach, we are able to improve analysis cost by 66%

   (a) Bioinformatics analysis consumes up to 90% of experiment costs [6]

   (b) Our approach reduces cost by enabling the use of clusters of smaller machines, which are more cost proportional [7]

2 Results

Architecture

1. ADAM provides a collection-oriented view of genomic data

   (a) Operations are built on Spark's Resilient Distributed Dataset (RDD, see [8]) abstraction, which provides a distributed, in-memory array

   (b) Collection-oriented view eliminates need for sort-order invariants; conflicting sort-order invariants make it difficult to chain current genomic analysis tools together

2. Processing pipelines are first class citizens in ADAM

   (a) Most genomic analyses are built as cascading pipelines

   (b) This has led to the design of many bioinformatics-specific workflow engines (e.g., Galaxy [9])

   (c) Spark is natively designed to support chaining analysis steps with stronger guarantees (fault tolerance, recomputation) by building a DAG [2]

   (d) Build *workflows*, not *workflow engines*

Analyses

1. Variant Calling on 1000 Genomes
2. TCGA variant analysis

2.0.1 Scalable Variant Calling

1. ADAM enables both:
   (a) Rapid variant calling of a single sample
   (b) Joint variant calling across a large cohort

2.0.2 TCGA Variant Analysis

1. Variants can impact transcriptional regulation [10, 11]
2. However, to study these impacts, we need to integrate across several types of genomic data, which is difficult in conventional pipelines
3. In ADAM, this analysis is trival to do
4. We demonstrate how this can be done using the somatic variant callset from The Cancer Genome Atlas [12]

3 Conclusion

TBA

References

1. McKenna, A. *et al.* The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research* **20,** 1297–1303 (2010).
2. Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S. & Stoica, I. *Spark: cluster computing with working sets* in *Proceedings of the USENIX Conference on Hot Topics in Cloud Computing (HotCloud '10)* (2010), 10.
3. Dean, J. & Ghemawat, S. MapReduce: simplified data processing on large clusters. *Communications of the ACM* **51,** 107–113 (2008).
4. Freeman, J. *et al.* Mapping brain activity at scale with cluster computing. *Nature Methods* **11,** 941–950 (2014).
5. Palsson, B. & Zengler, K. The challenges of integrating multi-omic data sets. *Nature chemical biology* **6,** 787–789 (2010).
6. Andrews, K. R. & Luikart, G. Recent novel approaches for population genomics data analysis. *Molecular Ecology* **23,** 1661–1667. ISSN: 1365-294X (2014).
7. Janapa Reddi, V., Lee, B. C., Chilimbi, T. & Vaid, K. *Web search using mobile cores: quantifying and mitigating the price of efficiency* in *Proceedings of the IEEE/ACM International Symposium on Computer Architecture (ISCA '10)* **38** (ACM, 2010), 314–325.

8. Zaharia, M. *et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing* in *Proceedings of the USENIX Conference on Networked Systems Design and Implementation (NSDI '12)* (2012), 2.

9. Goecks, J., Nekrutenko, A., Taylor, J. & The Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol* **11,** R86 (2010).

10. Weingarten-Gabbay, S. & Segal, E. The grammar of transcriptional regulation. *Human genetics* **133,** 701–711 (2014).

11. Levo, M. & Segal, E. In pursuit of design principles of regulatory sequences. *Nature Reviews Genetics* **15,** 453–468 (2014).

12. Weinstein, J. N. *et al.* The cancer genome atlas pan-cancer analysis project. *Nature genetics* **45,** 1113–1120 (2013).

Methods

4 Architecture

1. In prior work [13], we have described the design approach that inspired ADAM [2]
2. ADAM uses a commodity columnar storage format to store genomic data
   (a) Use the Apache Parquet [14] storage format, loosely based off of Google Dremel [15]
   (b) ADAM's formats are fully compatible with legacy genomics file formats

   (c) Parquet enables compression that is between BAM and CRAM [16] without incurring the costs and restrictions of reference-based compression

   (d) Additionally, we are able to apply these same compression techniques to variant data, leading to a representation that is 66% smaller than GZIP-ed VCF

   (e) This data is queryable from a variety of query engines (Spark-SQL [17], Impala [18])

3. Many reference-based genomics algorithms can be implemented using the “region join” abstraction

   (a) At it’s base, a “region join” is a relational join where key equality is determined by the similarity of two keys in a genomic region space

   (b) This primitive can be used to implement many feature analysis algorithms [19]

   (c) To enable efficient implementations, we back our “region join” with a variety of execution engines (e.g., broadcast, shuffle-based sort)

5 Dataset Hosting

1. To make it easier to use ADAM, we’ve developed the eggo tool that ingests datasets

   (a) Dataset descriptions are stored using Data Protocols style schema [20]

   (b) Datasets are converted and stored in Amazon S3 as a public resource

   (c) Tool makes it easy for people to use ADAM on common genomic datasets

2. We use eggo to drive all of the analyses in this paper

6 Variant Calling Pipeline

1. Our variant calling pipeline is similar to the GATK's best practices pipeline [21]

Alignment

1. Short reads are aligned using the SNAP aligner [22]
2. Alignment is parallelized within avocado
3. To improve parallel efficiency of alignment, we:
    (a) Add custom code to perform multithreaded alignment index load from HDFS
    (b) Run SNAP as a daemon per machine, which allows amortization of index load costs

Read Preprocessing Algorithms

1. Run duplicate marking and base quality score recalibration
2. Details TBA; will talk about:
    (a) Efficient ways to parallelize quality score table aggregation
    (b) Benefits of fragment structure for duplicate marking

Variant Calling Algorithm

1. We use the avocado variant caller, which is built on ADAM, to call variants

2. avocado uses a local reassembly based approach for identifying variants

    (a) We make use of a complexity filtering process to limit the amount of reference regions that are locally reassembled [23]

    (b) Reassembler is de Bruijn graph based [24]

3. avocado uses a biallelic statistical model

    (a) Based off of the statistical model in SAMtools mpileup [25]

    (b) To improve numerical precision when integrating across large datasets, our statistical model is evaluated in log space

7 Regulatory Variant Analysis

1. Use "regulatory grammar" described by Weingarten et al [10]

2. Approach:

    (a) Ingest TCGA somatic variant calls

    (b) Region join against ENCODE data and featurize variant calls into regulatory region modifications

3. Final analysis "slices and dices" in two ways:

    (a) Cluster genes

    (b) Cluster on samples

    (c) More details TBA

References

13. Nothaft, F. A. *et al. Rethinking Data-Intensive Science Using Scalable Analytics Systems* in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '15)* (2015).

14. Apache. *Parquet* `http://parquet.incubator.apache.org`.

15. Melnik, S. *et al.* Dremel: interactive analysis of web-scale datasets. *Proceedings of the VLDB Endowment* **3,** 330–339 (2010).

16. Fritz, M. H.-Y., Leinonen, R., Cochrane, G. & Birney, E. Efficient storage of high throughput DNA sequencing data using reference-based compression. *Genome Research* **21,** 734–740 (2011).

17. Armbrust, M. *et al. Spark SQL: Relational Data Processing in Spark* in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '15)* (2015).

18. Kornacker, M. *et al. Impala: A Modern, Open-Source SQL Engine for Hadoop* in *Proceedings of the Conference on Innovative Data Systems Research (CIDR '15)* (2015).

19. Quinlan, A. R. & Hall, I. M. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* **26,** 841–842 (2010).

20. Open Knowledge Foundation Labs. *Data Protocols* `http://dataprotocols.org/`.

21. Auwera, G. A. *et al.* From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline. *Current Protocols in Bioinformatics,* 11–10 (2013).

22. Zaharia, M. *et al.* Faster and more accurate sequence alignment with SNAP. *arXiv preprint arXiv:1111.5572* (2011).

23. Bloniarz, A. *et al. Changepoint Analysis for Efficient Variant Calling* in *Research in Computational Molecular Biology* (2014), 20–34.

24. Pevzner, P. A., Tang, H. & Waterman, M. S. An Eulerian path approach to DNA fragment assembly. *Proceedings of the National Academy of Sciences* **98,** 9748–9753 (2001).

25. Li, H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27,** 2987–2993 (2011).

Acknowledgements The authors would like to thank their many colleagues who provided feedback on early implementations of ADAM, and on drafts of the manuscript. This research is supported in part by a National Science Foundation CISE Expeditions Award [grant number CCF-1139158], Lawrence Berkeley National Lab [grant number 7076018], a Defense Advanced Research Projects Agency XData Award [grant number FA8750-12-2-0331], a National Institutes of Health BD2K Award [grant number 1-U54HG007990-01], a National Institutes of Health Cancer Cloud Pilot Award [grant number HHSN261201400006C] and gifts from Amazon Web Services, Google, SAP, The Thomas and Stacey Siebel Foundation, Adatao, Adobe, Apple, Inc., Blue Goji, Bosch, C3Energy, Cisco, Cray, Cloudera, EMC, Ericsson, Facebook, Guavus, Huawei, Intel, Microsoft, NetApp, Pivotal, Samsung, Splunk, Virdata, VMware, and Yahoo!. Author FAN is supported by a National Science Foundation Graduate Research Fellowship.

Competing Interests The authors declare that they have no competing financial interests.

Correspondence Correspondence and requests for materials should be addressed to F.A.N. (email: fnothaft@berkeley.edu).