SeqWare Query Engine: Storing and Searching Sequence Data in the Cloud

Brian D O'Connor [1], Barry Merriman [2], Stanley F Nelson*[2]

[1]UNC Lineberger Comprehensive Cancer Center, University of North Carolina, Chapel Hill, NC, USA
[2]Department of Human Genetics, University of California, Los Angeles, CA, USA

Email: Brian D O'Connor - brianoc@email.unc.edu; Barry Merriman - barrym@ucla.edu; Stanley F Nelson*- snelson@ucla.edu;

*Corresponding author

Abstract

Background: Since the introduction of next-generation DNA sequencers, the rapid increases in sequencer throughput have caused the cost of human genome sequencing to drop dramatically. As a result, more than a dozen human genomes have been resequenced and published based on this technology, and these efforts are merely the first steps towards a future in which genome resequencing will be commonplace for biomedical research and clinical applications. The sharp increase in sequencer output strains all facets of the computational analysis workflow, especially the databasing and querying of the results. Future genomics projects need to represent tens, hundreds, or thousands of genomes in a database environment, which must also be scalable to keep pace with escalating data production. The advent of cloud computing and a variety of powerful tools designed to process petascale datasets provide an ideal solution to these ever increasing demands. In this work, we present a prototype cloud-based query engine designed to support these future needs, and compare its performance to more traditional databasing solutions.

Results: We present the implementation of a query engine using the highly scalable, NoSQL HBase database backend from the Hadoop project. This project was used to store and query the full range of variants and annotations determined by the recent sequencing of the U87MG human cancer cell line. The query engine integrates with the SeqWare Pipeline tools which were used to trigger alignments, variant calling, and annotate coding consequences and dbSNP statuses for the U87MG Single Nucleotide Variants (SNVs) and indel variants. In addition to the HBase backend, we created a web-based frontend that provides both an interactive query

interface and also integrates with widely used genome browsers and analysis tools. Using this new tool, users can query variants (SNVs, indels, translocations, etc) with a rich level of annotations such as coverage, functional annotation, and variant type. In addition to the U87MG genome, we loaded additional whole genome sequence variants and a glioblastoma multiforme tumor/normal pair to both profile backend performance and also provide examples of using the Hadoop MapReduce framework with the SeqWare Query Engine. This software is open source and freely available from http://seqware.sourceforge.net and the sample U87MG whole genome database can be queried from http://genome.ucla.edu/U87.

Conclusions: The SeqWare Query Engine provided an easy way to make the U87MG genome sequencing project available to non-programmers and programmers alike. This enabled quicker and more open exploration of the results, faster tuning of parameters for various heuristic filters designed to optimize variant calling accuracy, and a common interface to the variant data to simplify development of analysis tools. The performance and scalability of HBase compared to Berkeley DB was favorable, particularly when coupled with MapReduce framework. The SeqWare Query Engine provides a good fit to the ever-growing genome sequence datasets because of the range of data types supported, the ease of querying and integrating with existing tools, and the robust scalability of the underlying cloud-based technologies.

Background

Recent advances in sequencing technologies have led to a greatly reduced cost of sequencing [1]. The dramatic costs reductions have shaped the experiments scientists have been able to perform and have opened up the possibility of whole human genome resequencing becoming commonplace. Currently over a dozen human genomes have been completed, most using one of the short read, high-throughput technologies that are responsible for this sequence growth [2–16]. The datatypes produced by these sequencing projects are varied, but most report single nucleotide variants (SNVs), small insertions/deletions (indels, typically <10 bases), structural variants (SVs), and may include additional information such as haplotype phasing and novel sequence assemblies. Paired tumor/normal samples can additionally be used to identify somatic mutation events by filtering for those variants present in the tumor but not the normal.

In this work we introduce the SeqWare Query Engine, a scalable database system intended to represent the full range of data types common to whole genome and other experimental designs for next generation

sequence data. HBase was chosen as the underlying backend because of its robust querying abilities using the Hadoop MapReduce environment and its auto-sharding of data across a commodity cluster based on the Hadoop HDFS distributed filesystem (http://hadoop.apache.org). We also present a web service that wraps the use of MapReduce to allow for sophisticated queries of the database through a simple web interface. The web service can be used interactively or programmatically and makes it possible to easily integrate with genome browsers, such as the UCSC Browser [17], GBrowse [18], or IGV (http://www.broadinstitute.org/igv), and with data analysis tools, such as the UCSC table browser [19], GALAXY [20], and others. The backend and web service can be used together to create databases containing varying levels of annotations, from raw variant calls and coverage to highly annotated and filtered SNV predictions. This flexibility allows the SeqWare Query Engine to scale from raw data analysis and algorithm tuning through highly annotated data dissemination and hosting. The design decision to move away from traditional relational databases in favor of the NoSQL-style of limited, but highly scalable, databases allows us to leverage the project to support tens of genomes now, and hundreds to thousands of genomes in the future, limited only by the underlying cloud resources.

Results

U87MG Genome Database

For our original U87MG human genome cell line sequencing project, we created the SeqWare Query Engine database, first built on BerkeleyDB (http://www.oracle.com/technetwork/database/berkeleydb) and then later ported to HBase (http://hbase.apache.org). For the new project presented here, the U87MG database was enhanced with the addition of the 13 other human genomes that were publicly available when this effort commenced. To further enhance the query engine, we also added new query strategies and utilities, such as a MapReduce-based variant search tool. Unlike the U87MG genome, which included all variant calls regardless of quality, the other genomes included only post quality filtered variants. Still, they offer a proof of concept that the HBase backend can represent multiple genomes worth of sequence variants and associated annotation data. This sample query engine is hosted at http://genome.ucla.edu/U87 and can be used for both programmatic and interactive queries through the web service interface. A database snapshot is not available for download but all the source datasets are publicly available and the database can be reconstituted in another location using either the BerkeleyDB or HBase backends along with the provided query engine loading tools.

Performance

Figure 1 shows a comparison between the BerkeleyDB and HBase backends for both variant load and variant export functions. BerkeleyDB was chosen as the original backend for multiple reasons: it did not require a database daemon to run, it provided a key-value store similar to the distributed key-value NoSQL stores we intended to move towards, it had a well-designed API, and it was known to be widely used and robust (for example, BerkeleyDB was used by the MySQL relational database as a supported database engine through the 5.0 MySQL release, http://www.mysql.com). In the load tests both BerkeleyDB and HBase performed comparably (Figure 1a), with both tests using a single thread with a similar client API (rather than a MapReduce loader which would be possible only with HBase). BerkeleyDB is slightly faster until about 6 million (6M) variants are loaded but HBase is faster after that point and eventually takes about 15 minutes less to load all 7M variants in this test. The test for variant export had a significantly different outcome (Figure 1b). Both MapReduce and standard single-thread API retrieval of variants from HBase were extremely efficient, with the MapReduce exporter completing in 45 seconds and the single-thread in 386 seconds. This is in sharp contrast with the BerkeleyDB backend which took 6,281 seconds to export the 7M variant records. When the BerkeleyDB database reached approximately 3.5M variants the run time to export ballooned quickly. This was likely due to memory limitations on the single node serving the BerkeleyDB, resulted in greatly degraded query performance once the index could no longer fit in memory. In contrast, the HBase cluster nodes were each responsible for storing and querying only a fraction of the data and this data sharding resulting in much more robust query performance. This clearly illustrates the scalability advantage of the HBase/Hadoop implementation.

Software Availability

The SeqWare Query Engine, like the rest of the SeqWare project, is fully open source and is distributed under the terms of the GNU General Public License v3 (http://www.gnu.org/licenses/licenses.html). The software can be downloaded from version control on the project's SourceForge.net site (http://seqware.sourceforge.net). Visitors to the site can also post questions on the mailing list, view documentation on wiki pages, and download a pre-configured SeqWare Pipeline and Query Engine as a virtual machine, suitable for running an a wide array of hardware. The present authors (BDO) are particularly interested in working with other developers on this project and welcome any contributions.

Discussion

If current trends are any indication, the falling costs of sequencing will soon make whole human genome datasets commonplace. Full genome sequence datasets, while increasingly common, are just one of many experimental designs that are currently used with this generation of sequencing platforms: whole exome sequencing, RNA sequencing (RNAseq), and bisulfite methylation sequencing are examples of other important genomic analyses that will require large scale databasing capabilities. Efforts such as the 1000 Genomes project (http://www.1000genomes.org), the Cancer Genome Atlas (TCGA, http://cancergenome.nih.gov), and the International Cancer Genome Consortium (http://www.icgc.org) are each generating a wide variety of such data across hundreds to thousands of samples.

The variety and number of sequencing datasets already produced, in production, or being planned present huge infrastructure challenges to the research community. Primary data, if available, are typically huge, difficult to transfer over public networks, and cumbersome to analyze without significant local computational infrastructure in the form of large compute clusters, extensive data storage facilities, dedicated system administrators, and bioinformaticians adept at low-level programming. Highly annotated datasets, such as finished variant calls, are more commonly available, particularly for human datasets. These present a more compact representation of the most salient information, but are typically only available as flat text files in a variety of quasi-standard file formats that require substantial "emotional involvement" to properly reformat and process. In many cases, essential source information has been eliminated for the sake of data reduction, making recalculation impossible. These challenges, both in terms of file sizes, diverse formats, and limited data retention, can make writing generic analysis tools complex and difficult. To facilitate the integration of experimental types and increase tool reuse, a common mechanism to store the variant calls and other key information from sequencing experiments is highly desirable. Properly databasing this information enables both a common structure and a query interface to support powerful mining of sequence-derived information.

To date most biological database projects have focused on storage of heavily annotated model organism reference sequences. For example, efforts such as the UCSC genome databases [21], the Generic Model Organism Database's Chado schema [22], and the Ensembl database [23] all solve the problem of storing reference genome annotations in a complete and comprehensive way. The focus for these databases is proper representation of biological data types and genome annotations, but not on storing many thousands of genomes worth of variants relative to a given reference. While many biological database schemas currently in wide use could support tens or even hundreds of genomes worth of SNP calls, ultimately these

systems are limited by the resources of a single database instance. Since they focus on relatively modest amounts of annotation storage, loading hundreds of genomes worth of multi-terabyte sequencing coverage information, for example, would likely overwhelm these traditional database approaches. Yet the appeal of databasing next generation sequence data is clear since it would simplify tool development and allow for useful queries across samples and projects.

Recent innovations from search-oriented companies such as Google, Yahoo, Facebook, and LinkedIn provide compelling technologies that could potentially enable computation on petascale sequence data. Projects such as Hadoop and HBase, open source implementations of Google's MapReduce framework [24] and BigTable database [25] respectively, can be converted to powerful frameworks for analyzing next generation sequencing data. For example, MapReduce is based on a functional programming style where the basic methods available to perform analysis are a map phase, where data is transfomed from one form to another, and a reduce phase, where data is sorted and condensed. This fits well to fundamental sequence analysis computations such as alignment and variant calling. Already there are versions of analysis algorithms such as Crossbow (alignment, [26]) and GATK (variant calling and other tools, [27]) that make use of the MapReduce paradigm. The approach is not fundamentally different from other functional programming languages that have come before, but in this case the approach is combined in Hadoop with both a distributed filesystem (HDFS) and tightly coupled execution engine. These tools, in turn, form the basis of the HBase database system. Unlike traditional grid technologies—for example a sun grid engine computation cluster—the Hadoop environment automatically partitions large data files across the underlying cluster, and computations can then be distributed across the individual data pieces without requiring the analysis program to know where the data resides, or manage such information. The HBase database system—one of the most mature of the NoSQL database projects—takes advantage of this sharding feature and allows a database to be broken into distinct pieces across the underlying computer cluster. This enables the creation of much larger databases than can be supported in traditonal relational implementations which are constrained to run on a single database server, up to the scale of billions of rows (e.g. bases in a genome) and millions of columns (e.g. individual genomes). This is considerably larger than most database systems typically support, but is the correct scale needed to represent heavily annotated whole genome sequence data for future large scale biomedical research studies and clinical deployment to the broadest patient populations.

Conclusions

We have briefly introduced the an improved version of the open source SeqWare Query Engine, that uses an HBase backend and the MapReduce/Hadoop infrastructure to provide cloud-based scalability to the SeqWare sequence analysis project. The results here show the scaling benefits that result from this, even when creating a database of genomic variants for just 13 human genomes. The basic database functions, such as importing and exporting, are one to two orders of magnitude faster with HBase instead of BerkelyDB, and moreover, the highly nonlinear improvement in scaling is readily demonstrated at the critical point where the standard database server becomes saturated, whereas the HBase server maintains a proper distributed load as the data burden is increased. This fully cloud-oriented database framework is ideal for the creation of whole genome sequence/variant databases. It is capable of supporting large scale genome sequencing research projects involving hundreds to thousands of genomes as well as future large scale clinical deployments utilizing advanced sequencer technology that will soon involve tens to hundreds of thousands of genomes.

Methods

Design Approach

The HBase backend for SeqWare Query Engine is based on the increasingly popular design paradigm that focuses on scalability at the expense of full ACID compliance, relational database schemas, and a proper query language (as reflected in the name "NoSQL"). The result is that, while scalable to thousands of compute nodes, the overall operations permitted on the database are limited. Each records consists of a key and the value, which consists of one or more "column families" that are fixed at table creation time. Each column family can have many "labels" which can be added at any time, and each of these labels can have one or more "timestamps" (versions). For the query engine database, the genomic start position of each feature was used as the key while four column families served to represent the core data types: variants (SNVs, indels, SVs, and translocations), coverage, features (any location-based annotations on the genome), and coding consequences which link back to the variants entries they report on. The coverage object stores individual base coverages in a hash and covers a user-defined range of bases to minimize storage requirements for this data type. Additional column family labels are created as new genomes are loaded into the database, and timestamps are used to distinguish variants in the same genome at identical locations. Figure 2 shows an example row with two genomes' data loaded. This design was chosen because it meant identical variants in different genomes are stored within the same row, making comparisons

between genomes extremely fasting using MapReduce or similar simple, uniform operators (Figure 2a). The diagram also shows how secondary indexes are handled in the HBase backend (Figure 2b). Tags are a convenient mechanism to associate arbitrary key-value pairs with any variant object and support lookup for the object using the key (tag). When variants or other data types are written to the database, the persistence code identifies tags and adds them to a second table where the key is the tag plus variant ID and the value is the reference genomic table and location. This enables variants with certain tags to be identified without having to walk the contents of the entire table.

Datasets

Fourteen human genome datasets were chosen for loading into a common SeqWare Query Engine backend, see Table 1 for the complete list. Most datasets included just SNV, small indel, and a limited number of SV predictions. The U87MG human cancer cell line genome was used to test the load of large-scale, raw variant analysis data types. For this genome, SNVs, small indels, large deletions, translocation events, and base-by-base coverage were all loaded. For the SNV and small indels, any variant observed once or more in the underlying short read data were loaded, which resulted in large numbers of spurious variants (i.e. sequencing errors) being loaded in the database. This was done purposefully for two reasons: for this study, to facilitate stress testing the HBase backend, and for the U87MG sequencing project, to facilitate analysis algorithm development by given practical access to the greatest potential universe of candidate variants. In particular, the fast querying abilities of the SeqWare Query Engine enabled rapid heuristic tuning of the variant calling pipeline parameters, based on many cycles of filtering and assessment that in turn required extensive querying of this vary large U87MG candidate variant database.
A secondary dataset generated in our lab, the "1102 GBM" tumor/normal whole genome sequence pair, was used to compare the performance between the BerkeleyDB and HBase Query Engine backend types. This dataset, like the U87MG genome, included loading raw variant calls seen once or more in both backends in order to profile the load and query mechanism.

Programmatic Access

The SeqWare Query Engine provides a common database store interface that supports both the BerkleyDB and HBase backend types. This store object provides methods to read and write the full range of data types into and out of the underlying database. It supports queries that lookup all variants and filter by specific fields such as coverage or variant call p-value. The store can also query based on secondary indexes

where the secondary index typically being a tag (key-value pair). The underlying implementation for each store type (BerkeleyDB or HBase) is quite different but the generic store interface masks the difference from the various import and export tools available in the project. The store interface was used whenever possible in order to maximize the portability of the query engine and to make it possible to switch backends in the future.

Two lower-level APIs are available for querying the HBase database directly. The first is the HBase API, which the store object uses for most of its operations including filtering variants by tags. This API is very similar to other database interfaces and lets the calling client iterate and filter result sets. HBase also support the use of a MapReduce source and sink, which allow for database traversal and load respectively. This was used to iterate over all variants in the database as quickly as possible and to perform basic analysis tasks such as variant type counting. The speed and flexibility of the MapReduce interface to HBase make it an attractive mechanism to implement future functionality.

Web Service Access

The web service is built on top of the programmatic, generic store object and uses the Restlet Java API (http://www.restlet.org). This provides a RESTful web services [28] for the query methods available through the store. When loaded from a web browser, the web service uses XSLT and CSS to display a navigatable web interface with user-friendly web forms for searching the database. Three data types are supported: variants, coding consequence reports and per-base coverage. Variants can be searched by tag and also a wide variety of fields such as their depth of coverage. The tag field is used to store a variety of annotations and, in the U87MG database, this includes dbSNP status, mutational consequence predictions, and names of overlapping genes, among others. For the variant reports, the standard BED file type (http://genome.ucsc.edu/FAQ/FAQformat) is supported and the user can alternatively select to load the query result directly in the UCSC browser, the IGV browser, or as a list of non-redundant tags comprising the variant annotation. Coverage information can be queried only by location, and the result can be generated in WIG format (http://genome.ucsc.edu/FAQ/FAQformat), WIG with coverage averaged by each block, or loadable links for the UCSC and IGV browsers.

When queried programmatically, the web service returns XML result files. These contain enough metadata to construct queries as URLs which can then be used to return query results in standardized formats (BED, WIG, etc). This makes it easy to construct URLs that return results from within any programming language. The stateless, URL-based nature of RESTful web services such as this ensure that the widest

variety of programming tools and environments are compatible with this tool. Since every query is just a URL, they can be created from within a web browser using the user-friendly form fields and cut-and-pasted into another tool or script. These URLs can be shared over email, linked to in a publication, and bookmarked for later use, thereby providing a convenient, universally understood reference to the results.

Data Load Tools

Most of the genome datasets used in this project where limited to SNV, indel, and SV predictions. For those genomes, the variant information files that were available were loaded into the query engine using either standard file type loaders (GFF, key-value files, etc) or using custom annotation file parsers. The pileup file format was used to load both the U87MG and "1102 GBM" tumor/normal genome variants and coverage information. The variant loading tool supports a plugin interface so new annotation file types can be easily supported. This import tool is available in the query engine package and uses the generic store interface for loading information in database backend (see "Programmatic Access" for more information). The HBase store currently uses an API very similar to other database connection APIs, and is therefore not inherently parallel, although multiple loads can occur simultaneously.

Analysis Tools

Two prototype analysis tools were created for use with the U87MG and "1102 GBM" tumor/normal databases. First, a MapReduce variant query tool was written to directly compare the performance of the retrieval of variants from HBase using the API versus using MapReduce. This simple tool used the HBase TableInputFormat object and a MapReduce job to traverse all database rows. The second analysis tool created was a simple somatic mutation detector for use with the "1102 GBM" tumor/normal genome database. Again, the HBase TableInputFormat object was used to iterate over the variants from the tumor genome, evaluating each variant by user-specified quality criteria in the map step and identifying those that were present in the cancer but not in the normal. In the reduce phase the coverage at each putative somatic mutation location was checked and only those where the coverage was good and no normal sample variant was called where reported as putative somatic mutations.

Performance Measurement

Backend performance was measured for both the BerkeleyDB and HBase stores using both data import and export time as the metric. The "1102 GBM" tumor/normal genome was used in this testing, which

included enough variant calls (both true and spurious) to stress test both backends. A single threaded, API-based approach was taken for the load test with both the BerkeleyDB and HBase backends. Each chromosome was loaded in turn and the time taken to import these variants into the database was recorded. A similar approach was taken for the retrieval test except in this case the BerkeleyDB backend continued to use an API approach whereas the HBase backend used both an API and MapReduced approach. The export test was interleaved with the import test, wherein after a chromosome was loaded the variants where exported and both processes were timed. In that way we monitored both the import and export time as a function of overall database size.

The HBase tests were conducted on a 6 node HBase cluster where each node contained 8 2.4GHz Xeon CPUs, 24GB of RAM, and 6TB of hard drive space. Each node contained 16 map task slots and 4 reducer task slots. The BerkeleyDB tests were conducted on a single node with the identical hardware.

Authors contributions

BDO designed and implemented the SeqWare Query Engine. BM provided guidance on project goals and applications, and choices of annotation databases and analysis algorithms. SFN is the principal investigator for the U87MG genome sequencing project which supported the development of SeqWare.

Acknowledgements

This work was supported by grants from the NINDS (U24NS), the Dani Saleh Brain Tumor Fund, and the Henry Singleton Brain Tumor Fund. The authors would like to acknowledge Jordan Mendler for his contributions to the SeqWare Pipeline project and Hane Lee for her feedback on the SeqWare Query Engine tools.

References

1. Snyder M, Du J, Gerstein M: **Personal genome sequencing: current approaches and challenges**. *Genes & development* 2010, **24**(5):423.
2. Lander E, Linton L, Birren B, Nusbaum C, Zody M, Baldwin J, Devon K, Dewar K, Doyle M, FitzHugh W, et al.: **Initial sequencing and analysis of the human genome**. *Nature* 2001, **409**(6822):860–921.
3. Levy S, Sutton G, Ng P, Feuk L, Halpern A, Walenz B, Axelrod N, Huang J, Kirkness E, Denisov G, et al.: **The diploid genome sequence of an individual human**. *PLoS Biol* 2007, **5**(10):e254.
4. Wheeler D, Srinivasan M, Egholm M, Shen Y, Chen L, McGuire A, He W, Chen Y, Makhijani V, Roth G, et al.: **The complete genome of an individual by massively parallel DNA sequencing**. *Nature* 2008, **452**(7189):872–876.
5. Pushkarev D, Neff N, Quake S: **Single-molecule sequencing of an individual human genome**. *Nature biotechnology* 2009, **27**(9):847–850.

6. Wang J, Wang W, Li R, Li Y, Tian G, Goodman L, Fan W, Zhang J, Li J, Zhang J, et al.: **The diploid genome sequence of an Asian individual**. *Nature* 2008, **456**(7218):60–65.

7. Bentley D, Balasubramanian S, Swerdlow H, Smith G, Milton J, Brown C, Hall K, Evers D, Barnes C, Bignell H, et al.: **Accurate whole human genome sequencing using reversible terminator chemistry**. *Nature* 2008, **456**(7218):53–59.

8. McKernan K, Peckham H, Costa G, McLaughlin S, Fu Y, Tsung E, Clouser C, Duncan C, Ichikawa J, Lee C, et al.: **Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding**. *Genome research* 2009, **19**(9):1527.

9. Ahn S, Kim T, Lee S, Kim D, Ghang H, Kim D, Kim B, Kim S, Kim W, Kim C, et al.: **The first Korean genome sequence and analysis: full genome sequencing for a socio-ethnic group**. *Genome research* 2009, **19**(9):1622.

10. Kim J, Ju Y, Park H, Kim S, Lee S, Yi J, Mudge J, Miller N, Hong D, Bell C, et al.: **A highly annotated whole-genome sequence of a Korean individual**. *Nature* 2009, **460**(7258):1011–1015.

11. Drmanac R, Sparks A, Callow M, Halpern A, Burns N, Kermani B, Carnevali P, Nazarenko I, Nilsen G, Yeung G, et al.: **Human genome sequencing using unchained base reads on self-assembling DNA nanoarrays**. *Science* 2010, **327**(5961):78.

12. Ley T, Mardis E, Ding L, Fulton B, McLellan M, Chen K, Dooling D, Dunford-Shore B, McGrath S, Hickenbotham M, et al.: **DNA sequencing of a cytogenetically normal acute myeloid leukaemia genome**. *Nature* 2008, **456**(7218):66–72.

13. Mardis E, Ding L, Dooling D, Larson D, McLellan M, Chen K, Koboldt D, Fulton R, Delehaunty K, McGrath S, et al.: **Recurring mutations found by sequencing an acute myeloid leukemia genome**. *New England Journal of Medicine* 2009, **361**(11):1058.

14. Pleasance E, Stephens P, O'Meara S, McBride D, Meynert A, Jones D, Lin M, Beare D, Lau K, Greenman C, et al.: **A small-cell lung cancer genome with complex signatures of tobacco exposure**. *Nature* 2010, **463**:184–190.

15. Pleasance E, Cheetham R, Stephens P, McBride D, Humphray S, Greenman C, Varela I, Lin M, Ordóñez G, Bignell G, et al.: **A comprehensive catalogue of somatic mutations from a human cancer genome**. *Nature* 2010, **463**:191–196.

16. Clark M, Homer N, O'Connor B, Chen Z, Eskin A, Lee H, Merriman B, Nelson S: **U87MG decoded: the genomic sequence of a cytogenetically aberrant human cancer cell line** 2010.

17. Kent W, Sugnet C, Furey T, Roskin K, Pringle T, Zahler A, et al.: **The human genome browser at UCSC**. *Genome research* 2002, **12**(6):996.

18. Stein L, Mungall C, Shu S, Caudy M, Mangone M, Day A, Nickerson E, Stajich J, Harris T, Arva A, et al.: **The generic genome browser: a building block for a model organism system database**. *Genome research* 2002, **12**(10):1599.

19. Karolchik D, Hinrichs A, Furey T, Roskin K, Sugnet C, Haussler D, Kent W: **The UCSC Table Browser data retrieval tool**. *Nucleic acids research* 2004, **32**(Database Issue):D493.

20. Giardine B, Riemer C, Hardison R, Burhans R, Elnitski L, Shah P, Zhang Y, Blankenberg D, Albert I, Taylor J, et al.: **Galaxy: a platform for interactive large-scale genome analysis**. *Genome research* 2005, **15**(10):1451.

21. Rhead B, Karolchik D, Kuhn R, Hinrichs A, Zweig A, Fujita P, Diekhans M, Smith K, Rosenbloom K, Raney B, et al.: **The UCSC genome browser database: update 2010**. *Nucleic acids research* 2009.

22. Mungall C, Emmert D, et al.: **A Chado case study: an ontology-based modular schema for representing genome-associated biological information**. *Bioinformatics* 2007, **23**(13):i337.

23. Hubbard T, Aken B, Beal K, Ballester B, Caccamo M, Chen Y, Clarke L, Coates G, Cunningham F, Cutts T, et al.: **Ensembl 2007**. *Nucleic acids research* 2006.

24. Dean J, Ghemawat S: **MapReduce: Simplified data processing on large clusters**. *Communications of the ACM* 2008, **51**:107–113.

25. Chang F, Dean J, Ghemawat S, Hsieh W, Wallach D, Burrows M, Chandra T, Fikes A, Gruber R: **Bigtable: A distributed storage system for structured data**. *ACM Transactions on Computer Systems (TOCS)* 2008, **26**(2):4.
26. Langmead B, Schatz M, Lin J, Pop M, Salzberg S: **Searching for SNPs with cloud computing**. *Genome Biology* 2009, **10**(11):R134.
27. McKenna A, Hanna M, Banks E, Sivachenko A, Cibulskis K, Kernytsky A, Garimella K, Altshuler D, Gabriel S, Daly M, et al.: **The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data**. *Genome Research* 2010.
28. Fielding R: **Architectural Styles and the Design of Network-based Software Architectures**. *PhD thesis*, University of California 2000.

Figures

Figure 1 - Load and Query Performance

Comparisons of load and query times between the HBase and BerkeleyDB backend. (a) Load times for the "1102 GBM" tumor/normal genomes where compared between HBase and BerkeleyDB. Both used a single-threaded approach to better compare relative performance. Both perform similarly but over time the load times for BerkeleyDB increase faster than with HBase. (b) Comparison of querying the 1102 genome database between BerkeleyDB, HBase single threaded, and HBase using MapReduce. Beyond 3M variants BerkeleyDB query times increase dramatically while both query types for HBase perform linearly, with MapReduce consistently exhibiting the best performance.

Figure 2 - SeqWare Query Engine Schema

The HBase database is a generic key-value, column oriented database that pairs well with the inherent sparse matrix nature of variant annotations. (a) The primary table stores multiple genomes worth of generic features, variants, coverages, and variant consequences using genomic location within a particular reference genome as the key. Each genome is represented by a particular column family label (such as "variant:genome7"). For locations with more than one called variant the HBase timestamp is used to distinguish each. (b) Secondary indexing is accomplished using a secondary table per genome indexed. The key is the tag being indexed plus the ID of the object of interest, the value is the row key for the original table. This makes lookup by secondary indexes, "tags" for example, possible without having to iterate over all contents of the primary table.

Tables

Table 1 - Datasets

Fourteen whole genome datasets were loaded into the database, including the U87MG genome, with the March 2006 assembly of the human genome used as reference (NCBI36/hg18). Variant types (SNVs, small/large indels, SVs, etc) loaded and publication references are noted for each respective dataset. This table was adapted from Snyder *et al.* 2010.

Dataset	**Technology**	**SNVs & Indels**	**SV**	**Translocations**	**Reference**
European-Venter	Sanger	Y	Y	N	Levy *et al.* 2007
European-Watson	454	Y	Y	N	Wheeler *et al.* 2008
European-Quake	Helicos	Y	Y	N	Pushkarev *et al.* 2009
Asian	Illumina	Y	Y	N	Wang *et al.* 2008
Yoruban 18507	Illumina	Y	Y	N	Bentley *et al.* 2008
Yoruban 18507	SOLiD	Y	Y	N	McKernan *et al.* 2009
Korean	Illumina	Y	Y	N	Ahn *et al.* 2009
Korean-AKI	Illumina	Y	Y	N	Kim *et al.* 2009
3 human genomes	Complete Genomics	Y	Y	N	Drmanac *et al.* 2009
AML T/N	Illumina	Y	Y	N	Ley *et al.* 2008
AML genome	Illumina	Y	Y	N	Mardis *et al.* 2009
Melanoma	Illumina	Y	Y	N	Pleasance *et al.* 2010a
Lung cancer	SOLiD	Y	Y	N	Pleasance *et al.* 2010b
U87MG	SOLiD	Y	Y	Y	Clark *et al.* 2010