Managing Data from High-Throughput Genomic Processing: A Case Study

Toby Bloom
Broad Institute of MIT and Harvard
tbloom@broad.mit.edu

Ted Sharpe
Broad Institute of MIT and Harvard
tsharpe@broad.mit.edu

Abstract

Genomic data has become the canonical example of very large, very complex data sets. As such, there has been significant interest in ways to provide targeted database support to address issues that arise in genomic processing. Whether genomic data is truly a special case, or just another application area exhibiting problems common to other domains, is an as yet unanswered question. In this abstract, we explore the structure and processing requirements of a large-scale genome sequencing center, as a case study of the issues that arise in genomic data managements, and as a means to compare those issues with those that arise in other domains.

1. Overview

The Broad Institute high-throughput genome sequencing center currently produces genome sequence at the highest rate in the world. The sequencing laboratory is essentially a large manufacturing facility: it uses DNA samples as raw material and produces digitized sequence where other manufacturing facilities might produce widgets. We produce over 50 billion high-quality nucleotide base calls per year, each of which has multiple pieces of information associated with it. The amounts of data produced by sequencing, as well as the data maintained for tracking the process, and reporting on progress, raise significant challenges for informatics resources. The following sections provide background on the sequencing processes that must be tracked, and the type and size of the data produced. We then discuss the issues that arise in performing those functions, and compare them to data management to problems that arise in other domains.

Proceedings of the 30th VLDB Conference,
Toronto, Canada, 2004

2. Background : the Genome Sequencing Process

The sequencing process starts with a piece of DNA and produces from it a character string of A's.C's, T's and G's that represent the four nucleotide bases from which DNA is composed. During the sequencing process, an initial DNA sample undergoes a series of laboratory procedures that include steps such as cutting the DNA into many small pieces, replicating each of those pieces many times, and attaching flourescent dyes. In the final step, those dyes are detected by lasers, and a signal trace (a "read") is produced for each of those small pieces. Tracking the samples through the various lab procedures is essential for mapping the millions of sequence reads from the laser sequence detectors back to the original DNA samples. Tracking is also needed to troubleshoot laboratory problems and monitor lab performance.

Once the signal trace is produced, the informatics data acquisition begins. The data is processed through a pipeline that cleans and validates each read. Signal processing software analyzes the binary trace file from the lasers, and produces the string of As, Cs, Ts, and G's denoting the DNA sequence. Since this is a chemical process and not absolutely accurate, we also maintain quality data for the sequence: the intensity of the flourescence for each of the four dyes at regular intervals, and a certainty score for each base (actually four certainty scores are produced.) Each read is typically about 800 to 900 bases long, with about 650 of those bases usable for further processing. The lab tracking data is used to associate each read with its original sample. The sequence is checked to make sure it looks like it came from the correct organism and that it hasn't been confused with or contaminated by other material in the lab process. Numerous metrics are collected for monitoring the process and for maintaining reportable data. The sequence data is then organized for assembly.

Once enough sequence data for a genome has been collected, the assembly process can be started. In assembly, the read sequences are matched against each other, like a giant jigsaw puzzle, to figure out in what order the reads appear in the organism's genome. There can be 40 to 60 million reads or more used in a single assembly, with no a priori structure known among them. The reads cover the genome many times over, to ensure enough data to fit all the pieces together unambiguously, despite the errors present in the detection process and the many nearly identical but distinct sequences that typically occur within a genome. Not all of the reads will fit together, and some will match in multiple places due to repeat structure within the genome.

Following assembly, there is often a process known as finishing, which determines where there are gaps in the assembled sequence, or regions with insufficiently high quality, and orders additional lab work to fill in the holes. Then the cycle repeats. But since other work, like annotation, or SNP detection, or comparative genomics projects will have begun, we have a versioning problem for assemblies – and a corresponding merge problem as well. These introduce yet more data complexity.

3. Challenges in Processing Genome Sequence Data

In this section, we discuss the issues that arise in managing the high-throughput genome sequence data. We then compare these challenges to those found in other domains.

3.1 Data Structure and Data Set Size

There are several categories of data handled in the sequencing informatics process: one is the sequence data itself; a second is the lab process workflow, tracking and process data; yet a third is the operational oversight information; and finally we have tracking of biological samples. In addition, we maintain the structure of the assembled genome, and the finishing data that includes versions and updates of those assemblies.

3.1.1 Sequence Data

The Broad Sequencing Center produces over 50 billion bases a year. Since most processing is performed at the read level, not the base level, in most cases we're dealing with millions, not billions, of entities. However, there are many data points associated with each base: the value of the most probable base (A,C,T,G); four quality scores representing the likelihood of each of the four possible calls at that position; numerous intensity scores – the intensity of the signal for each of the four dye colors—in the vicinity of the intensity peak that marks the presence of a base.

The most probable base and the quality score associated with it are, by far, the most frequently used data, and these attributes are stored as a set of parallel strings, at the read level. Thus, one string contains the called bases. Another contains the certainty score for each of those bases, in order. This significantly reduces the overhead of storing the data. However, that comes at the expense of processing. All queries on that data now need to parse those strings.

The signal traces themselves (which include not only the base calls and quality scores, but the intensity data in four channels for each sample) are maintained in file systems in a hashed directory structure, indexed by the database. This obviously increases complexity of queries on that data, while significantly reducing data size.

The scale is not as large as one might expect. Compression techniques have reduced our storage requirements to only about 5 terabytes per year. Our database is currently only about 1 terabyte. But within that, we maintain multiple tables of hundreds of millions of rows each. The size of our indexes is therefore significant, and query optimization becomes more complex.

Perhaps surprisingly, transaction rates are not problematic. The transactions can be complex, primarily because they involve coordination with external equipment, but the rate is relatively low. The system handles approximately one million lab transactions per day, and another one million data analysis transactions. Most of the data within the sequence acquisition system is write-once, which significantly reduces the complexity of transaction management.

3.1.2 Assembly and Closure Data

One example of the complex structures we maintain is the assembly structure used in the closure and finishing process that follows assembly.

The process starts with the set of contiguous sequences generated from the assembly process. We maintain the map of these "contigs", along with their predicted location on the genome, information about gap sizes between contigs, where known, and the locations of all the reads within those stretches of sequence. We also maintain the map of all large-insert clones that span contiguous regions, but for which we don't yet have full sequence coverage. These templates can be used to generate additional sequence to fill gaps. During the iterative finishing process, we order additional sequencing of these spanners, and maintain the information about pending orders, the primers that were ordered for purchase for each those, and then the results of those new sequencing requests. New sequence generated must be placed on the assembly, thereby generating a new version, with different coordinates for each of the reads and the contigs. In fact, since the new sequence may close gaps, some contigs merge, and so we need to maintain the history of the assembly, so that when other orders come back, or work is performed by a collaborator on an earlier version, it can be mapped to the new assembly. In addition, neighboring contigs must be updated to reflect their new neighbors. All changes tend to propagate through the assembly, with the attendant naming and relationship problems. And this structure may identify thousands of gaps, thousands of spanners, and millions of reads. The relationship management and propagation problems present interesting performance issues.

3.1.3 Taxonomies

One of the issues that arises regularly is the representation of non-uniform hierarchies. We encounter a number of situations in which branches of the tree may be of varying lengths in what should be analogous situations. This makes it difficult to structure the database. Taxonomies are one such situation.

The sequencing center handles large numbers of samples. These need to be categorized with respect to taxonomic classifications, from the genus and species, sometimes down to the individual. We need to know when we're sequencing a single individual vs. samples from multiple different individuals. This hierarchy is not uniform. Genus and species are standard, but the taxonomic levels above them vary among the kingdoms. And below that level, there is also a lack of uniformity: . some organisms are classified as subspecies within that hierarchy; others have breeds, or strains. In some organisms, an isolate is a single individual; in others a colony. The DNA from inbred mice of the same strain might be considered interchangeable in some situations. This means that there is no single hierarchy that represents the taxonomy. A simple organism dimension for categorizing sequencing targets becomes problematic.

3.1.4 Sample Data and Derivations

Another example of a problem in representing hierarchies is the DNA sample tracking in the lab. DNA is sequenced by cutting it into pieces and sequencing the ends of the pieces. The problem arises because any strand of DNA can be cut into a set of smaller strands of DNA, until you get down to a piece you can read in its entirety. (The usual process, however stops far short of this exhaustive recursion, however. We assemble continuous sequence by stitching together overlapping bits of separate samples, rather than exhaustively sequencing a single sample.) However, depending upon the task at hand, different techniques and lengths are used.

Thus, we may take a sample of genomic DNA and create from it sequencing libraries of small clones – hence creating a two-level hierarchy to maintain. Or we may instead create a library of large insert clones. We might sequence the ends of those without further cutting, or we might instead create from each large-insert clone, its own small-insert library. And from small-insert libraries, we might create yet smaller "shatter" libraries. At each level, we can take one DNA entity and create a library of smaller pieces of DNA. Or at any level, we can sequence directly. We therefore have a recursive structure in the sample dimension. We need to maintain the relationship among all those levels – which samples at one level were used to derive sets of samples at another level, so that we can understand how the sequences generated are related. There is no flat, tabular layout that represents these relationships, because the depth can vary, and there are no fixed joins that can find all related samples.

3.2 Querying

Queries again pose an interesting contrast to business applications.

The hierarchical data structures and compression techniques described above will of course add to query complexity. Many of our reports perform the equivalent of tree-walks on very large tables, with large numbers of constraints. Tricks like nested sets help, but are not sufficient [2]. Many of our complex queries require hand optimization. Better automatic parallelization might be a substantial help.

We have not yet addressed lab requests for trending and various other time-series reports. These are likely to present significant challenges as well.

In addition, the databases serve as back ends for complex genome analysis applications. As such, the queries presented often expect result sets numbering in the millions or tens of millions of rows, pulled from tables with hundreds of millions of rows. Rather than expecting aggregated results from the database, these applications require the individual records. Performance in retrieving those results is frequently a problem.

3.3 Impact of a Research Environment

In most respects, the sequencing center is no more than a large, automated flexible manufacturing floor. It happens to generate digitized DNA sequence rather than widgets, but in other ways, it differs little.

However, it should not be forgotten that this is a research environment, and some aspects of that environment are responsible for the complexity of data handling, rather than the complexity being inherent in the data itself.

There is an explicit goal of changing the lab processes and technology every six months, with major upgrades every two years. From an informatics standpoint, that means the number of steps in the process, the lab measures maintained at each step, the kinds of branching and pooling that occur in the workflow must be fully flexible. The process is separated from the basic data acquisition as much as possible, but the schema still changes very frequently. Constraints become very hard to maintain and automated error checking is difficult, because so little information is embedded in the schema. Thus, some of the complexity of our schemas, and the resulting complexity and performance issues in our queries, are a result of designing for flexibility rather than performance.

3.4 Data Integration and Analysis

Thus far, we've described the issues involved in maintaining sequencing data in isolation. But of course, the goal is genetic analysis, not just production of sequence data. Integration problems are the problems most often discussed in bioinformatics. Our center alone collects not only sequence data, but expression analysis data, and genotyping data. We will soon be collecting proteomic data as well. Integrative genomics [5] involves combining results from analysis of multiple kinds of data where any one is insufficient. And of course, there are large numbers of public and proprietary databases available with overlapping but not identical sets of genomic data of various types. Often the naming across them is inconsistent. The formats are certainly inconsistent. And so we bring all of the well known problems of data integration to a domain with many, many terabytes of data, and on-demand integration. This may be a quantitative rather than qualitative difference from other application domains, but it will still be a very significant challenge.

4. Conclusions

This abstract describes many of the problems faced in managing large-scale genomic data. We have addressed issues of data structure, and its impact on transactions and on queries; the unique issues brought on by very large data size, and the complexity of the application space.

All of these issues present serious challenges to maintaining and accessing this type of data. We have illustrated no problems here that do not exist in other domains, in whole or in part. However, the conjunction of these problems, in a domain with such large data volume does present significant challenges. We summarize here the various issues we raised and assess their impact in other application domains.

Traversing complex data structures, and maintaining the complex hierarchical relationships among the data entities, is certainly one of the biggest underlying problems in this space. However, this is the same problem that led to so much work on object databases years ago. It is not new.

The frequently changing structure and the impact on the schema again occur at larger scales here. But that problem underlies the work on metadata repositories as well. These kinds of problems are not unlike those that occur in a flexible manufacturing environment, or in environments such as clinical trials, where there is a basic framework that must be customized for each use. The frequency, as well as the scale, is different here. But the problem is similar. There is a tradeoff between the generality needed to allow for unpredictable changes, and the complexity of querying and data validation in those environments.

Data integration across multiple sources on demand certainly requires significant work to become a reality in genomic analysis. The problems are identical to those faced in other domains: inconsistent naming, inconsistent ranges, differing data formats. The sheer size of the problem may appear to verge on a qualitative difference here, but integration is a major problem everywhere.

Overall, at least from the perspective of sequence data, which is a small subset of genomic data in general, we see many unresolved issues in data management, but none that seem unique to this type of application data.

Acknowledgements

Andrew Zimmer leads the finishing informatics effort and architected the mapping structures to meet the challenges described. J.P.Leger is leading the data warehousing effort and manages the reporting team. Jen Baldwin, Rob Nicol, Chad Nussbaum, Bruce Birren, and Kerstin Linblad-Toh direct the Genome Sequencing and Analysis platform and program. Jill Mesirov is the Chief Informatics Officer of the Broad Institute, and Eric Lander is Founding Director of the Broad Institute. And thanks go to all members of the Production Sequencing Informatics team for building the software that enables the sequencing center to function.

References

[1] S. Batzoglou, et al ARACHNE: A whole-genome shotgun assembler. *Genome Res.* 12: 177-189 (2002).

[2] J. Celko, SQL for Smarties, Academic Press, San Diego, 2000.

[3]International Human Genome Sequencing Consortium, Initial Sequencing and Analysis of the Human Genome, Nature, 409, 860-921, (2001).

[4]D.B. Jaffe, et al. Whole-genome sequence assembly for mammalian genomes: Arachne 2. *Genome Res.* 13: 91_96 (2003).

[5]V.K. Mootha, et. al . Identification of a gene causing human cytochrome c oxidase deficiency by integrative genomics. *Proc Natl Acad Sci USA* 100: 605_610 (2003).