Data-Driven Genomic Computing (GeCo): Making sense of Signals from the Genome

Stefano Ceri, Anna Bernasconi, Arif Canakoglu, Andrea Gulino, Abdulrahman Kaitoua, Marco Masseroli, Luca Nanni, and Pietro Pinoli

Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Milano, Italy
{firstname.lastname}@polimi.it

Abstract. Next Generation Sequencing is a 10-year old technology for reading the DNA, capable of producing massive amounts of genomic data - in turn, reshaping genomic computing. In particular, tertiary data analysis is concerned with the integration of heterogeneous regions of the genome; this is an emerging and increasingly important problem of genomic computing, because regions carry important signals and the creation of new biological or clinical knowledge requires the integration of these signals into meaningful messages. We specifically focus on how the GeCo project is contributing to tertiary data analysis, by overviewing the main results of the project so far and by describing its future scenarios.

Keywords: Genomic Computing · Data Translation and Optimization · Cloud Computing · Next Generation Sequencing · Open Data

1 Introduction

Genomics is a relatively recent science. The double helix model of DNA, due to Nobel prizes James Watson and Francis Crick, was published on Nature in April 1953 and the first draft of the human genome, produced as result of the Human Genome Project, was published on Nature in February 2001, with the full sequence completed and published in April 2003. The Human Genome Project, primarily funded by the National Institutes of Health (NIH), was the result of a collective effort involving twenty universities and research centers in the United States, the United Kingdom, Japan, France, Germany, Canada, and China.

In the last fifteen years, the technology for DNA sequencing has made gigantic steps. Figure 1 shows the cost of DNA sequencing in the last fifteen years; by inspecting the curve, one can note a huge drop around 2008, with the introduction of Next Generation Sequencing, a high-throughput, massively parallel technology based upon the use of image capturing [22]. The cost of producing a complete human sequence dropped to 1000 US$ in 2015 and is expected to drop below 100 US$ in the next two-three years.

Each sequencing produces a mass of information (raw data) in the form of "short reads" of genome strings. Once stored, the raw data of a single genome reach a typical size of 200Gbyte; it is expected that between 100 million and 2

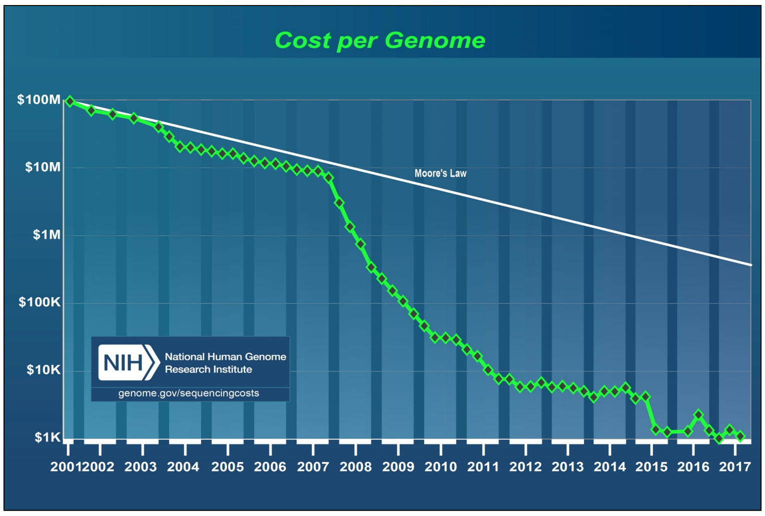

Fig. 1. Cost of DNA Sequencing, Source: NIH

billion human genomes will be sequenced by 2025, thereby generating the biggest "big data" problem for the mankind [23].

1.1 From strings to signals

Technological development also marked the generation of new methods for extracting signals from the genome, and this in turn is helping us in better understanding the information that the genome is bringing to us. Our concept of genome has evolved, from considering it as a long string of 3.2 billions of base pairs, encoding adenine (A), cytosine (C), guanine (G), and thymine (T), to that of a living system producing signals, to be integrated and interpreted. The most interesting signals can be classified as:

- mutations, telling us specific positions or regions of the genome where the code of an individual differs from the expected code of the "reference" human being. Mutations are associated with genetic diseases – which are inherited and occur on specific positions of the genes – and other diseases such as cancer – which are also produced during the human life and correlate with factors such as nutrition and pollution.
- gene expression, telling us in which specific conditions genes are active (i.e. they transcribe a protein) or inactive. It turns out that the same gene may have an intense activity in given conditions and no activity in others.
- peaks of expression, indicating specific positions of the genome where there is an increase of short reads due to a specific treatment of DNA; these in turn indicate specific biological events, such as the binding of a protein to the DNA.

These signals can be observed by using a genome browser, i.e. a viewer of the genome. All signals are aligned to a reference genome (the standard se-

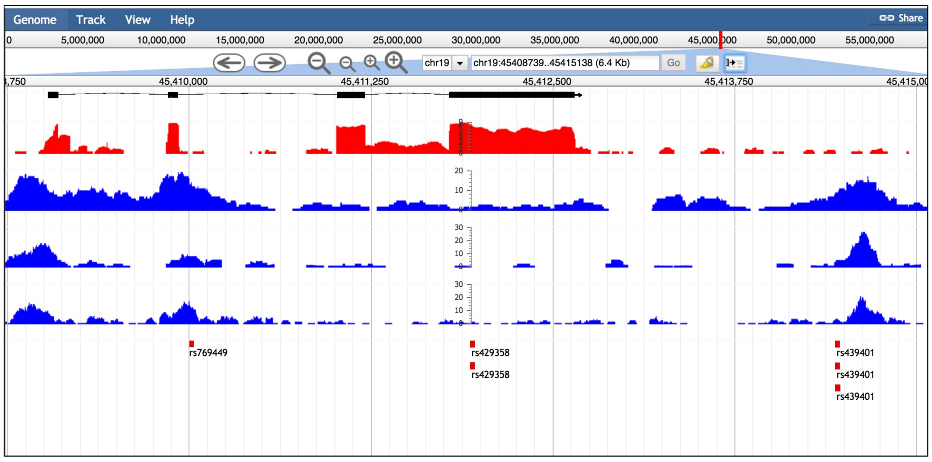

Fig. 2. Signals on a genome browser, corresponding to genes (an annotation in the header where genes are shown in black), gene expression (one track with red signal), peaks (three tracks with blue signals) and mutations (three tracks with red segments pointing to mutated positions of the genome)

quence characterizing human beings; such sequence is constantly improved and republished by the scientific community). The browser is open on a window of a given length (from few bases to millions of bases) and the signals are presented as tracks on the browser; each track, in turn, displays the signal – either by showing its intensity or just by showing its position.

Figure 2 presents a genome browser window; tracks of different colours describe gene expression, peaks of expressions, and mutations. The black line indicates the position of (four) genes – these are known information, or "annotations", that can be included in the window. An interesting biological question could be: "find the genes which are expressed where there are three peaks (representing the fact that the specific event denoted by the peak is confirmed by all experiments) and such that the gene is close to at least one mutation". Such question would, in this specific example, extract the second gene; GeCo's objective is to allow the expression of such questions over genomic signals.

1.2 Tertiary Data Analysis

Signals can be loaded on the browser only after being produced as a result of long and complex bioinformatics pipelines. In particular, analysis of NGS data is classified as primary, secondary, and tertiary (see Figures 3 and 4): primary analysis is essentially responsible of producing raw data - i.e. small sequences in output from the sequencing machine; secondary analysis is responsible of extracting ("calling") the signals from raw data and aligning them to the reference genome; tertiary analysis is responsible of data extraction, integration, and analysis. Thousands of processed datasets are being produced at large sequencing centers, are being assembled by large international consortia, and made available

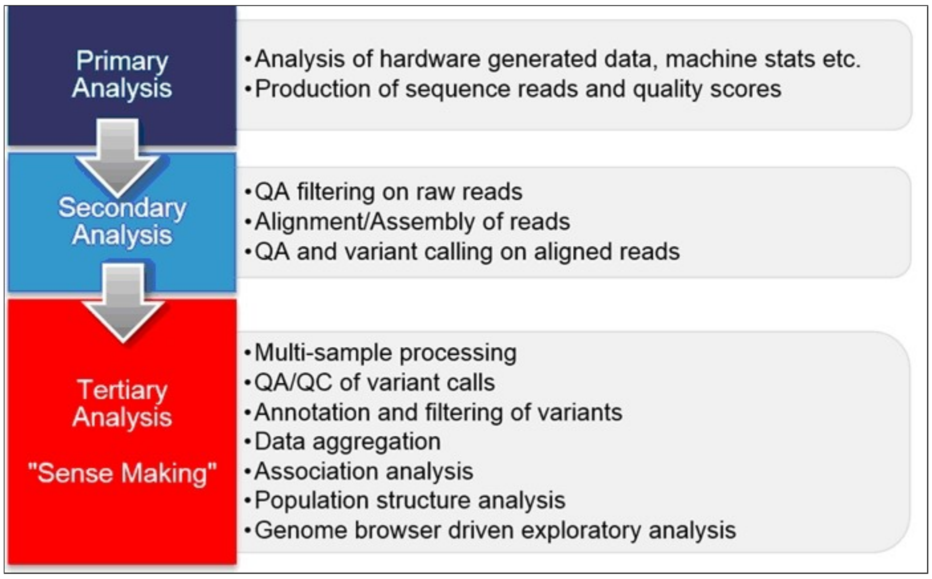
Primary Analysis
•Analysis of hardware generated data, machine stats etc.
•Production of sequence reads and quality scores
Secondary Analysis
•QA filtering on raw reads
•Alignment/Assembly of reads
•QA and variant calling on aligned reads
Tertiary Analysis
"Sense Making"
•Multi-sample processing
•QA/QC of variant calls
•Annotation and filtering of variants
•Data aggregation
•Association analysis
•Population structure analysis
•Genome browser driven exploratory analysis

Fig. 3. Primary, secondary, and tertiary data analysis for genomics, from `https://goo.gl/ko6pCE`

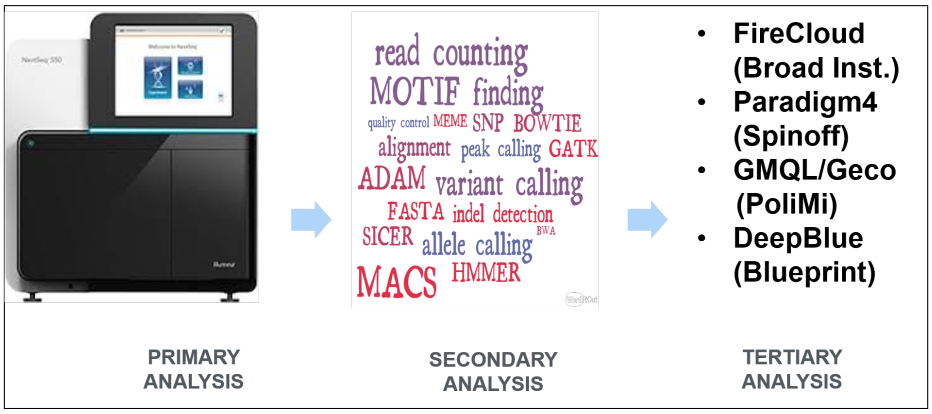

Fig. 4. The landscape of genomic computing tools; few of them are dedicated to tertiary data analysis

for secondary research use. International consortia include the Encyclopedia of DNA Elements (ENCODE) [12], The Cancer Genome Atlas (TCGA) [24], and the 1000 Genomes Project [1].

While many bioinformatics systems are dedicated to secondary data analysis, a few of them include tertiary analysis. Among these: SciDB, a scientific database produced by the spin-off company Paradigm4, supports a genomic addition focused on genomic data integration [3]; DeepBlue, provides easy access to datasets produced within the BluePrint consortium, with a language which is quite similar to GeCo's query language GMQL [2]; FireCloud, developed by the Broad Institute, offers an integrated platform supporting cancer research pipelines [13]. All these systems already support access to a huge number of open datasets, including ENCODE and TCGA.

2 Overview of GeCo

The introduction has set the stage for discussing why GeCo is an important project in the context of tertiary data analysis for genomics. We now illustrate the main results achieved by the project, providing as well a short description of the various publications where such results are presented.

2.1 Data Model

The Genomic Data Model (GDM) [18] is based on the notions of datasets and samples and on two abstractions: one for genomic regions, which represent portions of the DNA and their features, and one for their metadata. Datasets are collections of samples and each sample consists of two parts: the region data, which describe the characteristics of genomic features called during secondary analysis, and the metadata, which describe general properties of the sample.

Genomic region/feature data describe a broad variety of molecular aspects, which are individually measured, and therefore they are available in a variety of formats which hamper their integration and comprehensive assessment. GDM provides a schema to the genomic features of DNA regions; thus, it makes such heterogeneous data self-describing and interoperable. We map data from data files in their original format into the GDM format when they are used, without including them into a database, so as to preserve the possibility for biologists to work with their usual file-based tools. The provided data schema has a fixed part, which guarantees the comparability of regions produced by different kinds of processing, and a variable part reflecting the "feature calling process" that produced the regions and describing the region features determined through various processing types.

Metadata characterize the high heterogeneity of genomic data and their processing; they are collected in a broad variety of data structures and formats that constitute barriers to their use and comparison. To cope with the lack of agreed standards for metadata, GDM models metadata simply as free arbitrary semi-structured attribute-value pairs, where attributes may have multiple values

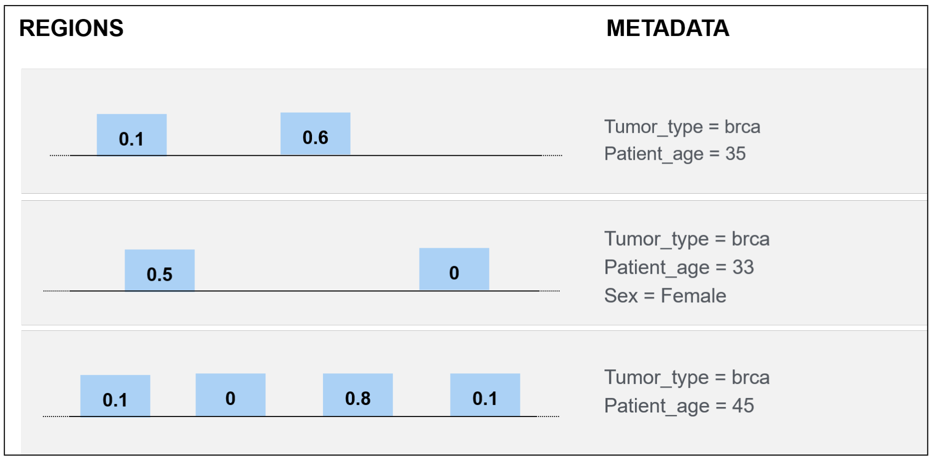

Fig. 5. Genomic Data Model

(e.g., the Disease attribute can have both "Cancer" and "Diabetes" values). We expect metadata to include at least the considered organism, tissue, cell line, experimental condition (e.g., antibody target – in the case of NGS ChIP-seq experiments, treatment, etc.), experiment type, data processing steps, feature calling, and analysis method used for the production of the related data; in the case of clinical studies, also individual's descriptions including phenotypes.

Figure 5 shows a GDM dataset consisting of three samples, each associated with a patient affected by Breast Cancer (BRCA). The region part has a simple schema, describing a particular value associated to each region; regions are aligned along the whole genome and belong to specific chromosomes (not shown in the figure). The metadata part includes free attribute-value pairs, describing the patient's age and sex; in GDM, attributes are freely associated to samples, without specific constraints.

2.2 Query Language

The name 'GenoMetric Query Language' (GMQL) [17] stems from the language's ability to deal with genomic distances, which are measured as nucleotide bases between genomic regions (aligned to the same reference) and computed using arithmetic operations between region coordinates. The language supports a very rich set of predicates describing distal conditions, i.e. distal properties of regions (e.g., being at minimal distance above a given threshold from given genes). Furthermore, the management of stranded regions (i.e. regions with an orientation) is facilitated by predicates that deal with such orientation (e.g., to locate the region's start and stop coordinates according to the strand, or the upstream or downstream directions with respect to the region's ends).

GMQL operations include classic algebraic operations (SELECT, PROJECT, UNION, DIFFERENCE, JOIN, SORT, AGGREGATE) and domain-specific operations (e.g., COVER deals with replicas of a same experiment; MAP refers genomic signals of experiments to user selected reference regions; GENOMETRIC

JOIN selects region pairs based upon distance properties); the full description and language specification of GMQL is provided at the GMQL website[1]. A typical GMQL query starts with a SELECT operation, which creates a dataset with only the data samples filtered out from an input dataset by using a predicate upon their attributes. Then, the query proceeds by processing the selected samples in batch with operations applied on their region data and/or metadata. Finally, it ends with a MATERIALIZE operation, which stores a dataset by saving the region data of each of its samples and the related metadata. Tracing metadata provenance during the processing of each operation is a unique aspect of our approach; knowing metadata associated with resulting samples allows tracing the input samples which contributed to results.

2.3 Implementations

The development of GMQL V1, reported in [10] was relatively fast, as the implementation took about one year; Pig Latin [19] was chosen as target language because GMQL syntactically recalls Pig Latin - both languages progressively construct queries by introducing intermediate results and naming them by using variables; this style of query writing was considered similar to the scripting style which is dominant in bioinformatics, although of course GMQL scripts are high-level and not intertwined with programming language constructs.

The main problem with this approach is that the optimization strategies were hard-coded within the software produced by the translator, which in turn was focused on the specific features of our target language Pig Latin. Moreover, at the end of 2014, it became clear that Apache Pig was no longer strongly supported, while other engines were becoming much more popular; in particular Spark [6] was achieving a dominant position, Flink [4] had very interesting capabilities as a streaming engine, and SciDB [21] had a totally different approach to storage through arrays. Therefore, at the beginning of 2015, we opted for a major system redesign, starting the development of GMQL Version 2.

The design of GMQL V2 [16] has been centered around the notion of an intermediate representation for the query language, developed as an abstract operator tree (actually a directed acyclic graph or DAG, as operations can be reused), that is at the center of the software architecture. The intermediate representation carries the semantics of the query language in terms of elementary operations that are applicable to metadata and to region data separately, and opens up to various options for expressing the language's syntax (which can be expressed as relational expressions, or in embedded form within programming or workflow languages, or in logical format by using a Datalog-like style) and for deploying over different cloud-based engines. In particular, at various times we used Spark, Flink, and SciDB.

[1] http://www.bioinformatics.deib.polimi.it/GMQL/

2.4 Comparison of Cloud-based Implementations

Compared to V1, V2 has a much more complex software organization. While V1 was cooperatively developed by 3 PhD students, V2 has a larger group of developers, which included in the past several master students; the use of GitHub allowed for several branches to the main Spark implementation, which allowed us to test also the use of Flink and SciDB, served by suitable implementation branches. At the moment, the Spark implementation is fully supported, while the Flink and SciDB implementations are only maintained for specific operations.

Comparative analysis, published in [8] and [9], shows that the performances of Flink and Spark are remarkably similar, while the performances of Spark and SciDB are very different, with SciDB being faster than Spark when operations involve selections and aggregates (as they are facilitated by an array organization), whereas Spark is faster than SciDB in JOIN and MAP operations (thanks to the general power of the Spark execution engine).

2.5 Data Indexing

While many solutions for efficient data management of "sequence reads" have been developed, in GeCo we concentrated on efficient data management for genomic intervals. GMQL is grounded on the efficient execution of operations for the composition and comparison of (epi)genomic regions and their associated attributes. To cope efficiently with complex region calculus, we have developed the 1D Interval Inverted Index (Di3), a multi-resolution single-dimension indexing framework [14].

Di3 is independent from the data layer and adaptable to any key-value pair persistence technology. These may range from NoSQL databases to classical B+ tree and simple in-memory key-value collections implemented by most of modern programming languages; such separation makes Di3 adaptable to a variety of application scenarios, from small scale ad-hoc solutions (using B+ tree), to large scale systemic solutions (using cloud-based key-value pair persistence technologies).

The most important contribution of Di3 is its extendable, orthogonal, and comprehensive region calculus. Region-based operations include the identification of co-occurrences or accumulations of regions, possibly from different biological tests and/or of distinct semantic types, within the same area of the DNA, sometimes within a certain distance from each other and/or from DNA regions with known structural or functional properties.

2.6 Desktop Data Analysis

We developed a rich set of abstractions for client-side data analysis, exploration and visualization, supported by a tool named GenoMetric Space Explorer (GeMSE) [15]; GeMSE supports primitives for data exploration spanning from *select*, *sort*, and *discretize*, to *clustering*, and *pattern extraction*. GeMSE leverages on GMQL as its back-end tertiary data retrieval framework, but can be used on any text files in standard BED (Browser Extensible Data), BroadPeak, NarrowPeak, GTF (General Transfer Format), or general tab-delimited format.

GeMSE applies to a *genometric space* produced by a specific GMQL operation, called `MAP` [17], which applies to two datasets, denoted as *reference* and *experiment*; the former typically corresponds to genes or transcription regulatory regions; the latter consists of multiple, possibly heterogeneous, samples, each constituted by multiple regions. The result is a matrix structure, called *genometric space*, where each row is associated with a reference region, each column refers to a sample, and each matrix entry is computed by means of an aggregate function (computed on a selected attribute of the experiment regions which overlap with the reference regions). GeMSE data exploration consists of three iterative phases:

– *Transition*, where a transformation function is applied on a genometric space resulting in a new genometric space;
– *Analysis*, where a genometric space is analyzed using data analysis functions (e.g., pattern analysis or statistical inference);
– *Visualization*, where a genometric space is visualized (e.g., on heatmaps or graph views).

Tracking multiple transformations of genometric spaces is crucial for data exploration. GeMSE tracks such transitions in a graph data structure called *State-Transition Tree* (STT), whose nodes represent different genometric spaces and whose edges represent the transformations between genometric spaces. From any data exploration state, one can view the related genometric space, visualizing it as a table or a heatmap, and also explore patterns contained into the heatmap. STT visualization facilitates state examination with data exploration and a trial-and-error approach.

3 Future Steps

3.1 Language Interfaces

Several activities are ongoing in GeCo. We fully developed a Python library supporting an interface to the full GMQL language; the library is deployed within the international Python library repository and does not require any installation - as it is customary in Python. It supports a local and a remote execution mode, the former runs in the client desktop, the latter runs on a remote server and as such it connects to the global repository. Similar interfaces are being deployed for R and Galaxy.

3.2 Distributed System

The availability of a core data model as a data interoperability solution and of a high-level data processing language is a strong prerequisite for defining data exchange protocols. We expect that each data repository will be the owner of the data that are locally produced, and that nodes of cooperating organizations will be connected to form a federated database. In such systems, queries move from a requesting node to a remote node; they are locally executed; then results are communicated back to the requesting node. This paradigm allows for distributing the processing to data, transferring only query results which are usually small in size.

Supporting a high-level query interface to a server is already making one big step forward, which is similar to the gigantic step made by SQL in the context of client-server architectures (which dates a couple of decades). Indeed, once a system supports an API for submitting GMQL queries, these have the following properties: they are short texts and produce short answers. In contrast, most of today's implementations are centralized and require first the full data transmission and next the evaluation of server-side programs. This scenario opens up to the design of simple interaction protocols, typically for:

- Requesting information about remote datasets, facilitated by the availability of metadata (for locating data of interest) and of their region schemas (for formalizing queries);
- Transmitting a query in high-level format and obtain data about its compilation, not only limited to correctness, but including also estimates of the data sizes of results;
- Launching query execution and then controlling the transmission of results, so as to be in control of staging resources and of communication load.

3.3 Repository

Very large-scale sequencing projects are emerging; as of today, the most relevant ones include:

- The Encyclopedia of DNA elements (ENCODE) [12], the most general and relevant world-wide repository for basic biology research. It provides public access to more than 4,000 experimental datasets, including the just released data from its Phase 3, which comprises hundreds of epigenetic experiments of processed data in human and mouse;
- The Cancer Genome Atlas (TCGA) [24], a full-scale effort to explore the entire spectrum of genomic changes involved in human cancer;
- The 1000 Genomes Project [1], aiming at establishing an extensive catalogue of human genomic variations from 26 different populations around the globe;
- The Roadmap Epigenomics Project [20], a repository of “normal” (not involved in diseases) human epigenomic data from NGS processing of stem cells and primary ex vivo tissues.

Data collected in these projects are open and public; all the Consortia release both raw and processed data, but biologists in nearly all cases trust the processing, which is of high-quality and well controlled and explained. The use of a high-level model and language, such as GDM and GMQL, is the ideal setting for provisioning next generation services over data collected and integrated from these and other repositories, improving over the current state-of-the-art. We already started to work towards an integrated repository: in [7] we discussed a conceptual representation of metadata, where we presented a minimal conceptual schema that includes data typically found in all platforms, albeit with different names and formats; in [11] we discussed the transformation of TCGA datasets into BED format, which is quite similar to GDM.

We are working towards the development of an integrated repository with the following features:

– Processed datasets available in several sources, including the above ones, will be provided with compatible metadata;
– Processed region datasets of other sources (beyond TCGA) will be translated to GDM;
– It will be possible to choose among a set of custom queries, representing the typical/most needed requests;
– It will be possible to provide user input samples to the services, whose privacy will be protected;
– Deferred result retrieval will be possible, through limited amount of staging at the sites hosting the services.

4 Conclusions

The progress in DNA and RNA sequencing technology has been so far coupled with huge computational efforts in primary and secondary genomic data management. In this paper, we have shown that a new need is emerging: making sense of data produced by these methods, in the so-called tertiary analysis; this is the objective of GeCo, our five-year project. In this paper, we have shown how GeCo is providing a new focus on data extraction, querying, and analysis by raising the level of abstraction of models, languages, and tools.

Acknowledgment. This research is funded by the ERC Advanced Grant project GeCo (Data-Driven Genomic Computing), No. 693174, 2016-2021.

References

1. 1000 Genomes Consortium. An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491, 56-65, 2012.
2. F. Albrecht et al. DeepBlue epigenomic data server: programmatic data retrieval and analysis of the epigenome. *Nucleid Acids Research*, 44(W1), W581-586, 2016.
3. Anonymous paper, Accelerating bioinformatics research with new software for big data to knowledge (BD2K), Paradigm4 Inc., 2015 downloaded from: `http://www.paradigm4.com/`).
4. Apache Flink. `http://flink.apache.org/`
5. Apache Pig. `http://pig.apache.org/`
6. Apache Spark. `http://spark.apache.org/`
7. A. Bernasconi et al. Conceptual modeling for genomics: building an integrated repository of open data. In: *Proc. Entity-Relationship*, Valencia, ES, 2017.

8. M. Bertoni et al. Evaluating cloud frameworks on genomic applications. *Proc. IEEE Conference on Big Data Management*, Santa Clara, CA, 2015.
9. S. Cattani et al. Evaluating big data genomic applications on SciDB and Spark. *Proc. Web Engineering Conference*, Rome, IT, 2017.
10. S. Ceri et al. Data management for heterogeneous genomic datasets. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 14(6), 1251-1264, 2016.
11. F. Cumbo et al. TCGA2BED: extracting, extending, integrating, and querying The Cancer Genome Atlas. *BMC Bioinformatics*, 18(6), 1-9, 2017.
12. ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489(7414), 57-74, 2012.
13. FireCloud. `https://software.broadinstitute.org/firecloud`
14. V. Jalili et al., Indexing Next-Generation Sequencing data, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.08.085.
15. V. Jalili et al. Explorative visual analytics on interval-based genomic data and their metadata. *BMC Bioinformatics*, 18, 536, 2017.
16. A. Kaitoua et al. Framework for supporting genomic operations, *IEEE-TC*, 10.1109/TC.2016.2603980, 2016.
17. M. Masseroli et al. GenoMetric Query Language: a novel approach to large-scale genomic data management. *Bioinformatics*, 31(12), 1881-1888, 2015.
18. M. Masseroli et al. Modeling and interoperability of heterogeneous genomic big data for integrative processing and querying. *Methods*, 111, 3-11, 2016.
19. C. Olston et al. Pig Latin: A not-so-foreign language for data processing. *ACM-SIGMOD*, 1099-1110, 2008.
20. Romanoski, C. E., et al. 2015. Epigenomics: Roadmap for regulation. Nature 518, 314-316.
21. SciDB. `http://www.scidb.org/`
22. S. C. Schuster. Next-generation sequencing transforms today's biology. *Nature Methods*, 5(1), 16-18, 2008.
23. Z. D. Stephens et al. Big data: astronomical or genomical? *PLoS Biology*, 13(7), 2015.
24. J. N. Weinstein et al. The Cancer Genome Atlas Pan-Cancer analysis project. *Nature Genetics*, 45(10), 1113-1120, 2013.
25. M. Zaharia et al. Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In *Proc. USENIX*, 15-28, 2012.