Volume 4, Issue 4, April 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

High-Throughput Genome Data Processing and Real-Time Analysis using Oracle Coherence In-Memory Technology

Sushma R. Vhatkar, Sanchika A. Bajpai
Department Of Computer Engineering
BSIOTR (W), Pune, India

Abstract— Today large sequencing centres are producing genomic data at the rate of 10 terabytes a day and require complicated processing to transform massive amounts of noisy raw data into biological information. With more and more medical knowledge discovered about genomic dispositions, the analysis and evaluation of genome data becomes a significant bottleneck in course of personalized medicine. This paper explores the potential and the limitations of using relational database systems as the data processing platform for high-throughput genomics. In this work, we present a completely new architecture for processing and analyzing genome data is builds on the in-memory database technology to eliminate time-consuming file-based data operations and to enable real-time data analysis. We found out that the use of in-memory technology as an integral component for genome data processing and its analysis significantly reduces time and costs to obtain relevant results.

Keywords— Real-Time Data Analysis; In-Memory Database Technology; Genome Data; Personalized Medicine; Next-Generation Sequencing

I. INTRODUCTION

Genomics has revolutionized almost every aspect of life sciences in the past decade. At the same time, technological advancement such as next-generation sequencing is transforming the field of genomics into a new paradigm of data-intensive computing [1]. A large sequencing centre such as the Broad Institute of Harvard and MIT can produce 10 terabytes of genomic data each day. The flood of data needs to undergo complex processing that mines biological information from vast sets of small sequence reads while handling numerous errors inherent in the data.

Genetic analysis labs are facing a serious data management problem: The development of high-throughput gene sequencing instruments makes mass genomics feasible, but at the same time produces gigabytes of sequencing data per experiment that need to be stored, aligned and analyzed. While it took 10 years and $3B dollars to produce a first draft of the human reference genome, the current generation of sequencing instruments is able to sequence between 2 to 4 billion bases in only a few days [3][4]. The current approach to data management for high-throughput genomics is file-centric and involves a large number of separate files containing a text-format or a proprietary binary format. Most of these formats are insufficiently documented and do not include meta-data. Some file formats that include metadata, do not separate the data's representation from its conceptual data model. This makes data workflow management very complicated and the analysis becomes inefficient. The current file-centric approach simply does not scale to the terabyte needs of high throughput genomics.

On the other hand, traditional database technology also has shortcomings that prevent it from becoming commonly used in this scientific domain: Database systems are optimized for fast processing of small and precise business data records. In this paper, we are interested to find out whether we have achieved high-throughput and real-time analysis of Genome data using Oracle Coherence In-Memory technology. We study different scenarios from high-throughput genomics and investigate how current in-memory database technology can be used for efficient data management. In particular, we are interested in faster, fault tolerant, highly available and scalable application.

In our demonstration, we will present a working prototype system using in-memory database technology. We will compare our system with a baseline implementation using existing software tools to show the effectiveness of our quality control mechanisms. Finally, we will perform parallel processing of expensive operations and show the improved performance.

II. RESEARCH BACKGROUND AND RELATED WORK

The In-Memory Database (IMDB) technology has demonstrated major advantages in analyzing big enterprise data [5], [6]. We developed a specific platform that combines processing and analyzing of genomic data as a holistic process based on the feedback of researchers and clinicians using Oracle Coherence In-Memory Data Grid. Our architecture is designed to run on existing hardware instead of highly specialized hardware to be cost-efficient and to make use of existing hardware infrastructures. In our implementation, we are developing our application using Oracle Coherence In-Memory Data Grid, which is a data management system for application objects that are shared across multiple servers, require low response time, very high throughput, predictable scalability, continuous availability and information reliability.

Coherence is fast, it stores all data solely in memory. There is no need to go to disk. Objects are always held in their serialized form. Holding data in a serialized form allows Coherence to skip the serialization step on the server meaning that data requests only have one serialization hit, occurring when they are de-serialized on the client after a response. Writes to the database are usually performed asynchronously. Asynchronous persistence of data is desirable as it means Coherence does not have to wait for disk access on a potentially bottlenecked resource. Queries run in parallel across the data grid. This leverages the entire hardware cluster simultaneously. Coherence includes a second level cache that sits in process on the client. This is a analogous to a typical caching layer, holding on to some defined number of objects previously requested by the client.

Coherence is both faults tolerant and highly available. That is to say that the loss of a single machine will not significantly impact the operation of the cluster. The reason for this resilience is that loss of a single node will result in a seamless failover to a backup copy held elsewhere in the cluster. All operations that were running on the node when it went down will also be re-executed elsewhere. It is worth emphasizing that this is one of the most powerful features of the product. Coherence will efficiently detect node loss and deal with it. It also deals with the addition of new nodes in the same seamless manor.

Coherence is Scalable. Adding new machines to the cluster increases the storage capacity by a factor of 1/n, where n is the number of nodes. CPU and bandwidth capacity will obviously be increased too as machines are added. This allows the cluster to scale linearly through the simple addition of commodity hardware. There is no need to buy bigger and bigger boxes.

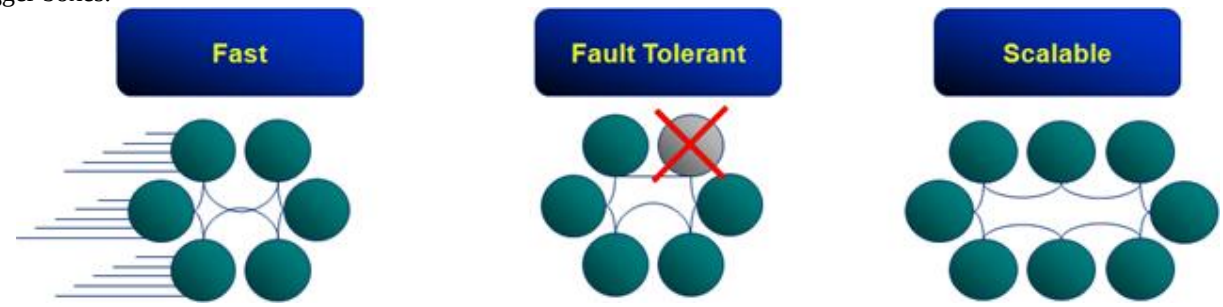

Fig. 1 Oracle Coherence properties

III. SYSTEM OVERVIEW

Our system for genome data processing and analysis works in an environment shown in Fig 2. Users of the system are hospitals and research institutes. A user starts by sending DNA samples to a third-party sequencing service and requests the produced genomic data to be transferred to our system, as shown by the arrow (1) in the figure. For each DNA sample, a sequencer produces raw images and then converts the image data to short read sequences (or reads, or brevity) of the genome. The read sequences are transferred to our system by shipping hard disks, as shown by the arrow (2). The data volume is usually hundreds of gigabytes per genome sample.

Once the data arrives at our system, the user can issue a request to process the data, including alignment of read sequences and detection of variations against a reference genome, as shown by the shaded box labeled as "II. Data Processing". The output of this module, including a whole genome sequence and variations detected, are stored in a database for further analysis. Afterwards the user can upload additional patient information, and initiate extensive analyses that combine genomic data and patient data. Such analyses are handled by the module labeled as "III. Deep Analysis", which automatically discovers patterns of both statistical significance and biological meanings.

A. Architecture

As an overall system architecture perspective, our research prototype consists of the architectural layers: data, platform, and application. In the following, all layers are described in detail. Fig. 3 depicts the system architecture of our system modeled as block diagram using the Fundamental Modeling Concepts.

1) Data Layer: The data layer holds genomic reference data, such as human reference genomes and annotations [7]. These data is the base for analysis of specific genomic findings. Additionally, it holds the patient-specific genomic data, which was generated by NGS devices. The latter needs to be analyzed in the course of personalized medicine and will be processed by the platform layer and combined by the applications of the application layer.

2) Platform Layer: The platform layer holds the complete process logic and the IMDB system for enabling real-time analysis of genomic data. In Fig. 3 on the right, our developed extensions of the platform layer, the worker and updater framework, are exemplarily depicted. The worker framework specifies for incoming sequencing request required tasks and subtasks and its order comparable to the map reduce approach [8]. It also dispatches these tasks to computing resource, such as computing odes, observes their status, and combines partial result sets to obtain the final result set. The updater framework is the basis for combining international research results. It regularly checks Internet ources, such as public FTP servers or web pages, for updated and newly added annotations, e.g. database exports or characteristic file formats, such as CSV, TXT, etc. New data is automatically downloaded and imported in the IMDB to extend the knowledge base. Once new data was imported, it is available for real-time analysis of genome data without any latency.

3) Application Layer: The application layer consists of special purpose applications to answer medical questions instead of generic purpose applications. Although these applications can only be used for a limited usecase, they are

highly optimized for solving these very specific tasks. All applications communicate via asynchronous Ajax calls and JavaScript Object Notation as data exchange format via a web service interface with the database layer [9], [10]. Accessing results or performing specific analysis is no longer limited o a single location, e.g. the desktop computer in the office of the physician. All application operations can be accessed from any device configured to have Internet access, which enhances productivity of its users.

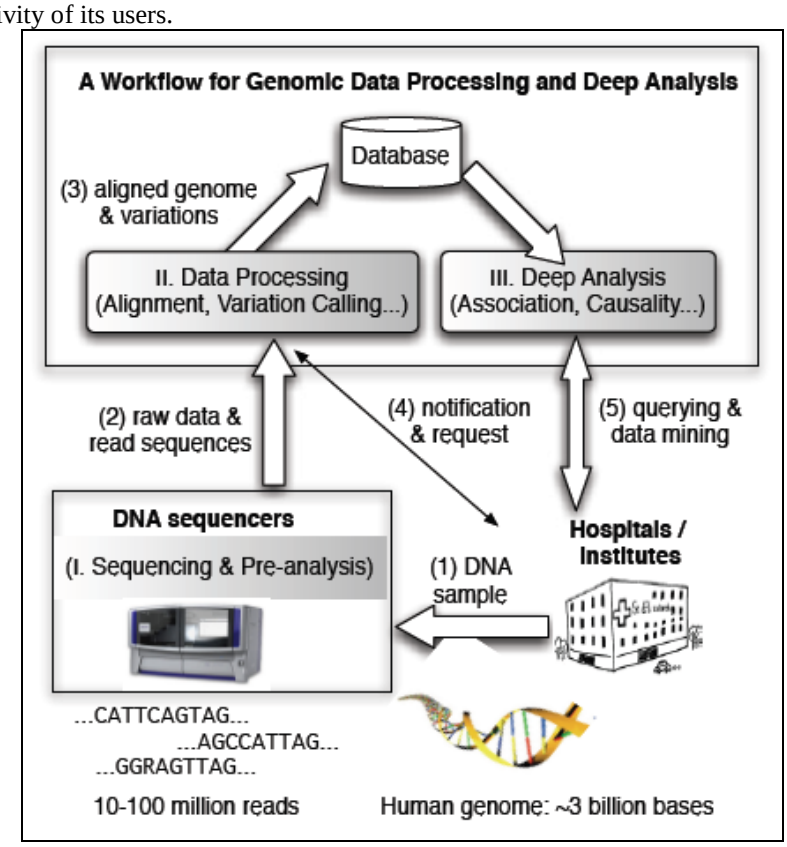

Fig. 2 Overall architecture.

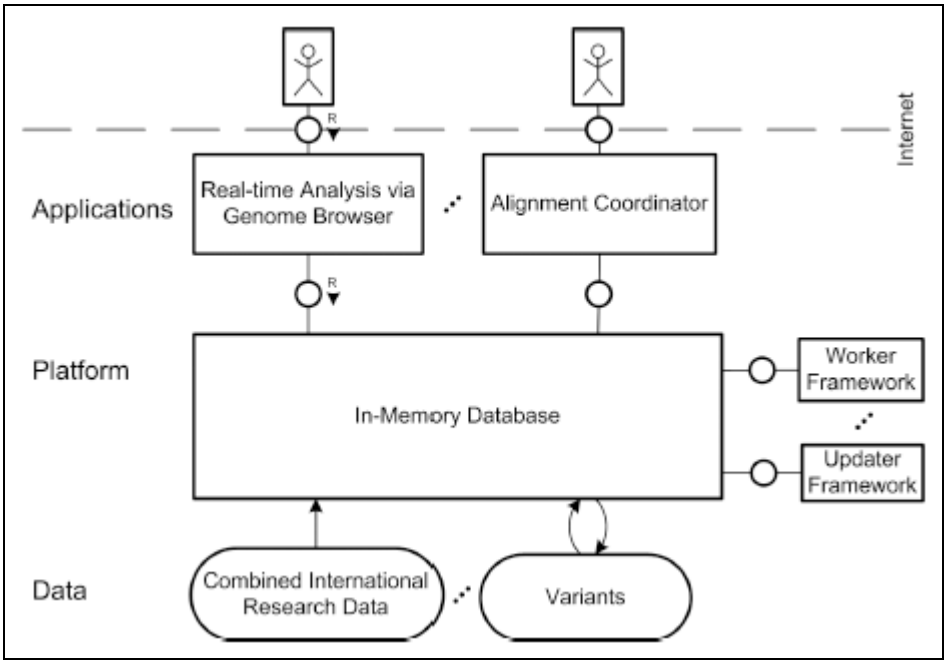

Fig. 3 The system architecture of our research prototype.

B. Data Modeling

As a first step, we want to develop a concise conceptual data model for gene sequencing experiments that is to be shared throughout all data processing phases. The goal is to identify all data entities and their relationships used in sequencing workflows. Important inputs for this design phase are example files for all workflow phases and a description of the expected analysis results. We used an implementation of the Entity-Relationship data model, the EDM, as a modeling tool [11].

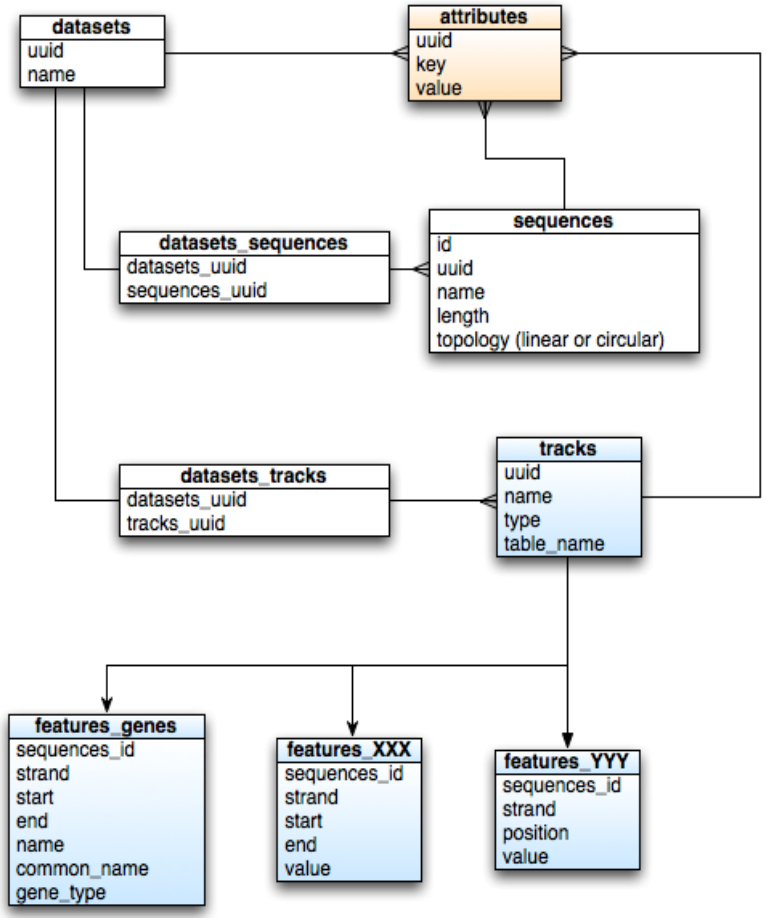

Fig. 4 Conceptual Data Model

C. Algorithm Development

Fig. 5 depicts a typical genome-processing pipeline as of today modeled as Business Process Modeling and Notation (BPMN)[12]. The integration of DNA in course of personalized medicine consists of the two major steps DNA sequencing and analysis of genome data. DNA sequencing spans the biological preparation of samples, e.g. blood or tissue, and its sequencing using a NGS device. The analysis of genome data is an IT-driven step processing FASTQ files from NGS devices, which includes alignment, variant calling, and the analysis of the results. Alignment is the reconstruction of the specific full genome by combining the acquired read sequences with a selected reference genome.

Variant calling detects anomalies in the reconstructed genome and checks whether these are possible variants, e.g. manifestation of certain alleles. The last and most time-intensive step is the analysis of all results from the variant calling and its interpretation using worldwide annotation databases. The genome browser of our HIG project addresses the ad-hoc analysis of the results without the need for time-consuming manual Internet searches.

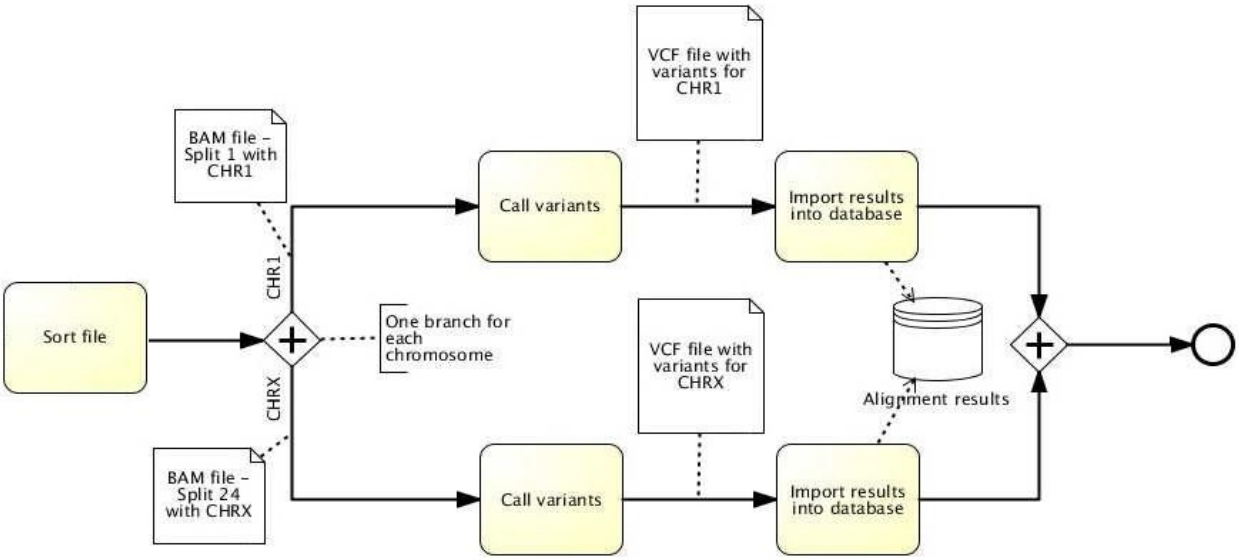

Fig. 5 Variant calling is processed in parallel

The inputs for alignment tasks are FASTQ files containing thousands or millions of raw DNA reads or snippets. FASTQ files are generated by the NGS device in a time intensive process. Instead of waiting for a single huge FASTQ file, we start processing as soon as possible, i.e. once FASTQ chunks, e.g. with a file size of 256MB, are generated by the NGS device. As a result, the data processing already starts while the sequencing run is still in progress. The results of the variant calling are stored in a task specific database table compatible to the Variant Calling Format (VCF) [13].

In the pipeline optimized for the IMDB technology the processing steps for sort, merge, and indexing are not performed by specific tools. These steps are directly executed by the IMDB without the need to create intermediate files in the file system.

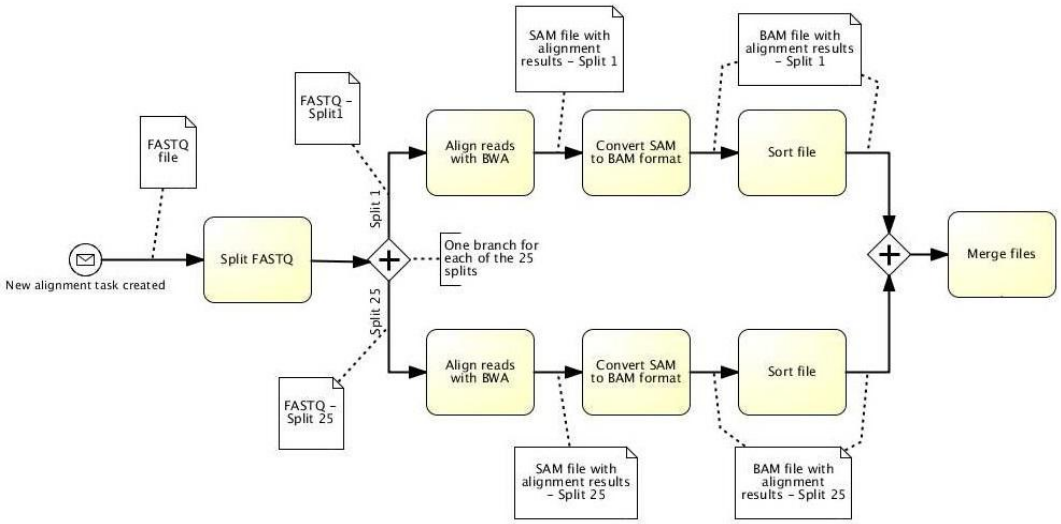

Fig. 6 Alignment algorithm which is called in parallel

IV. EVALUATION AND DISCUSSION

Our benchmarks verify two hypotheses: Firstly, the use of an IMDB as primary storage improves the overall execution time of established alignment algorithms, such as BWA, integrated in our High-Throughput Genome system architecture. Secondly, our system supports parallel execution of intermediate process steps across multiple compute nodes, which results in an additional performance improvement compared to the execution on a single compute node. The result incorporating 25 compute nodes show that the system was not completely loaded by our benchmarks since we did not adapt individual processing tools, which still implement single threaded execution models.

The best relative improvement shows the adapted pipeline using the IMDB as primary storage with at least 74 percent on single compute node and up to 89 percent on 25 compute nodes. It shows that the overall pipeline execution time correlates to the number of base pairs contained in the FASTQ file in a linear way. The scaling factor for the overall execution time varies between 1.80 and 1.96 across all experiments and file sizes. This indicates a very constant and predictable system behavior of our High-Throughput Genome system for varying input file size. For example, a doubled number of reads results in an overall response slightly below a factor of two. It helps to predict execution times and to supervise the correct system functionality.

Furthermore, our results stress the benefits of an IMDB for operating on intermediate results. The pipeline optimized for the IMDB replaces individual tools operating on files for specific process steps, such as sorting, merging, and indexing. In contrast, these operations are replaced by the IMDB operations without the need to create intermediate files in the File System (FS). We integrated existing alignment and variant calling tools into our High-Throughput Genome architecture without modifying their code. Thus, the speed-up documented in our benchmarks is mainly achieved by replacing selected file-based operations by IMDB operations.

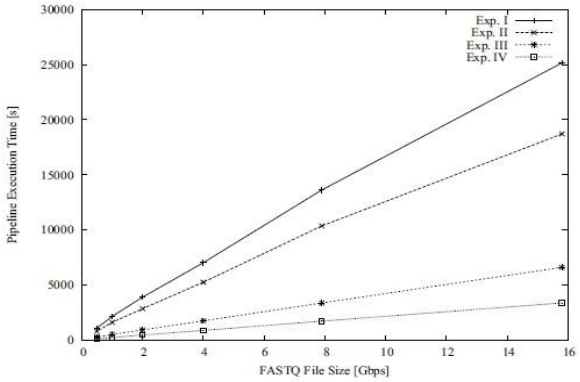

Fig. 7 Show improvements up 76% and 89% resp. on 25 compute nodes

V. CONCLUSION AND FUTURE WORK

In the given work, we presented our High-Throughput Genome system architecture for genome data processing based on an IMDB system. Based on our in-memory research activities, we outlined the applicability of this technology for processing of genome data to enable its real-time analysis. Furthermore, we shared insights in the specific architecture layers from an IT perspective and outlined details about select IMDB extensions for genome data processing, such as scheduling, worker framework or updater framework.

The obtained benchmark results showed that our High-Throughput Genome system architecture improves overall pipeline execution time by at least 25 percent on a single compute node and up to 89 percent involving 25 compute nodes. The performance boost is mainly derived from substituting intermediate processes, such as sorting, merging, and indexing by native IMDB operations. In future, we will investigate the impact of optimized alignment and variant calling algorithms that directly incorporate IMDB technology. We expect to further eliminate media breaks and to improve performance due to data proximity.

REFERENCES

[1] F. S. Collins et al., "New Goals for the U.S. Human Genome Project," *Science, vol. 282*, no. 5389, pp. 682– 689, 1998.
[2] W. J. Ansorge, "Next-generation DNA Sequencing Techniques," *New Biotechn, vol. 25*, no. 4, pp. 195–203, 2009.
[3] K. Jain, *Textbook of Pers. Medicine*. Springer, 2009.
[4] M.-P. Schapranow et al., "Mobile Real-time Analysis of Patient Data for Advanced Decision Support in Personalized Medicine," in *Proceedings of the 5th Int'l Conf. on eHealth, Telemedicine, and Social Medicine*, 2013.
[5] M.-P. Schapranow, *In-Memory Technology Enables History-Based Access Control for RFID-Aided Supply Chains*. Springer London, 2013, ch. 9, pp. 187–213.
[6] H. Plattner, *A Course in In-Memory Data Management: The Inner Mechanics of In-Memory Databases*. Springer, 2013.
[7] A. Knöpfel, B. Grone, and P. Tabeling, *Fundamental Modeling Concepts: Effective Communication of IT Systems*. John Wiley & Sons, 2006.
[8] S. Wandelt et al., "Data Management Challenges in Next Generation Sequencing," *Datenbank-Spektrum*, vol. 12, no. 3, pp. 161–171, 2012.
[9] S. Pabinger et al., "A Survey of Tools for Variant Analysis of Next-generation Genome Sequencing Data," *Brief. Bioinform*, Jan. 2013.
[10] K. Wang, M. Li, and H. Hakonarson, "ANNOVAR: Functional Annotation of Genetic Variants from Highthroughput Sequencing Data," *Nucleic Acids Res*, vol. 38, no. 16, 2010.
[11] V. Makarov, T. O'Grady, G. Cai, J. Lihm, J. D. Buxbaum, and S. Yoon, "Anntools," *Bioinformatics*, vol. 28, no. 5, pp. 724–725, Mar. 2012.
[12] The 1000 Genomes Project Cons., "A Map of Human Genome Variation from Population-scale Sequencing," *Nature*, vol. 467, no. 7319, pp. 1061—1073, Oct. 2010.
[13] H. Li and R. Durbin, "Fast and Accurate Short Read Alignment with Burrows-Wheeler Transformation," *Bioinformatics*, vol. 25, pp. 1754–1760, 2009.

