# High-Performance In-Memory Genome Project: A Platform for Integrated Real-Time Genome Data Analysis

Matthieu-P. Schapranow, Franziska Häger, Hasso Plattner
*Hasso Plattner Institute*
*Enterprise Platform and Integration Concepts*
*August–Bebel–Str. 88*
*14482 Potsdam, Germany*
*{schapranow|franziska.haeger|plattner}@hpi.uni-potsdam.de*

***Abstract*—A variety of next-generation sequencing technologies reduced costs and improved quality for whole genome sequencing within the last decade. However, interpretation and analysis of generated raw genome data is still a time- and resource-intensive task taking up to weeks. We applied in-memory database technology to form a completely new system architecture that enables processing and real-time analysis of genome data in a single system and reduces time and costs to obtain relevant results, e.g., in the course of personalized medicine.**

***Keywords-Genome Data Analysis; Process Integration; In-Memory Database Technology; Personalized Medicine; Next-Generation Sequencing.***

## I. Introduction

The Human Genome (HG) project officially launched in 1990 involved thousands of worldwide research institutes and required more than a decade to sequence and decode the full HG [1]. Next-Generation Sequencing (NGS) devices enable processing of whole genome data within hours while reducing costs [2]. NGS is used to support personalized medicine, which aims at treating patients specifically based on individual dispositions, e.g., genetic or environmental factors [3].

The In-Memory Database (IMDB) technology has proven to have major capabilities for analyzing big enterprise and medical data, e.g., to identify relevant patient data and to protect markets from injecting pharmaceutical counterfeits [4], [5].

In this work, we present our findings of applying IMDB technology to enable real-time analysis of genome data in course of our High-performance In-memory Genome (HIG) project. We developed a specific IT platform that combines processing and analyzing of genomic data as a holistic process based on the feedback of researchers and clinicians. Our HIG architecture is designed to run on commodity hardware instead of highly specialized hardware a) to be cost-efficient and b) to make use of existing hardware infrastructures. Fig. 1 depicts the system architecture of our HIG system modeled as block diagram using the Fundamental Modeling Concepts (FMC) [6].

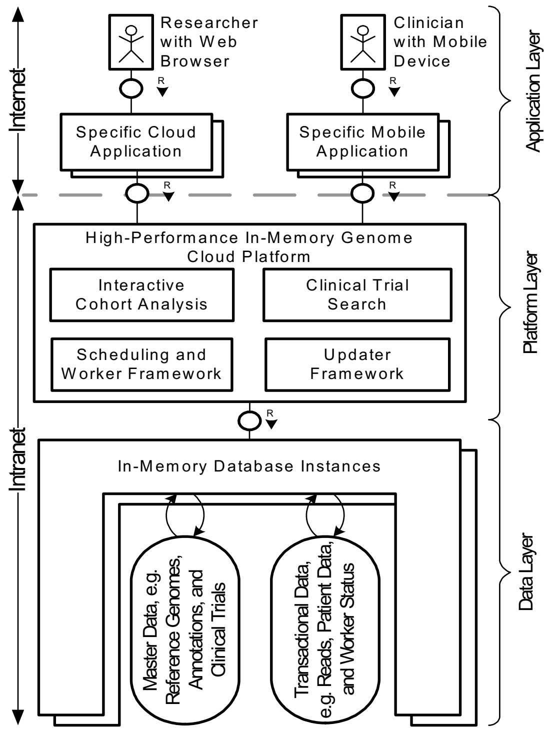

Researcher with Web Browser
Clinician with Mobile Device
R
Specific Cloud Application
Specific Mobile Application
Internet
Intranet
High-Performance In-Memory Genome Cloud Platform
Interactive Cohort Analysis
Clinical Trial Search
Scheduling and Worker Framework
Updater Framework
In-Memory Database Instances
Master Data, e.g. Reference Genomes, Annotations, and Clinical Trials
Transactional Data, e.g. Reads, Patient Data, and Worker Status
Application Layer
Platform Layer
Data Layer

Figure 1. The HIG system architecture consist of application, platform, and data layer. Analysis and processing of data is performed in the platform layer eliminating time-consuming data transfer.

The rest of the paper is structured as follows: In Sect. II, our work is set in context of related work. We introduce selected in-memory technology building blocks in Sect. III and present selected components of our HIG system in Sect. IV. In Sect. V, we share our benchmark results and discuss them in Sect. VI. Our work concludes with an outlook in Sect. VII.

## II. Related Work

Fig. 2 provides a comparison of costs for sequencing and main memory modules. Both costs follow a steadily declining trend, which facilitates the increasing use of NGS for whole genome sequencing and IMDB technology for its data analysis. Related work in the

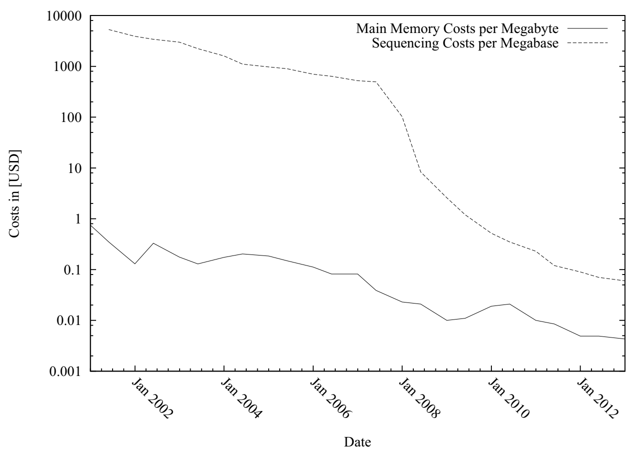


Figure 2. Costs for next-generation sequencing and main memory 2001-2013 adapted from [9], [10].

field of genome data processing has increased in the last years. However, work focusing on implementing end-to-end processes is still rare. Thus, our contribution focuses on supporting integrated processes.

Pabinger et al. evaluated workflow systems and analysis pipeline tools [7]. They realized that existing tools either miss flexibility or the end-user needs additional know-how to install them. We address this by introducing a combined system for modeling and execution of individual pipeline configurations without the need for command line scripts as presented in Sect. IV. Furthermore, Pabinger et al. analyzed a variety of variant analysis tools and evaluated their functionality. For web-based tools they see a drawback in the required data preparation before analysis because "[. . . ] files need to be packed, sorted and indexed before they can be used [7]". We address time-consuming data transformation and preparation steps and replace them by native operations directly performed in our incorporated in-memory database as discussed in Sect. VI.

Wandelt et al. have observed a trend towards more and more cloud-based NGS data management solutions [8, Sect. 4.3]. They identified the efficient mapping of workflow tasks in distributed computing environments and the adjustment of a given workflow to a dynamic environment as open issues. Our work contributes by providing a system architecture that combines processing and analyzing of genome data within a single system. Firstly, the worker framework developed in Python enables integration of computing resources across platform and Operating System (OS) borders. Secondly, the scheduler adjusts the execution of a given workflow, e.g., concrete pipeline process steps. It enables processing of multiple tasks in parallel, e.g., simultaneous users or multiple departments as described in Sect. IV.

## III. In-memory Technology Building Blocks

We refer to IMDB technology as a toolbox of IT artifacts to enable processing of enterprise data in real-time in the main memory of server systems [11]. The combination of IMDB database technology and analysis of genome data is driven by the declining cost trends as described in Sect. II. In the following, we outline selected building blocks of the IMDB technology and their relevance for real-time analysis of genomic data.

### A. Insert-only

Insert-only is a data management approach that stores data changes as new entries. In contrast to traditional databases destructive update or delete operations do not destroy the original data in an Insert-only table [12, Sect. 7.1]. Instead the data are invalidated, thus keeping complete history of value changes and the latest value for a certain attribute accessible [11]. This approach complies with legal regulations to permanently store clinical data and enables the tracing of decisions within the treatment process, e.g., to retrospectively perform analysis when certain treatments were initiated.

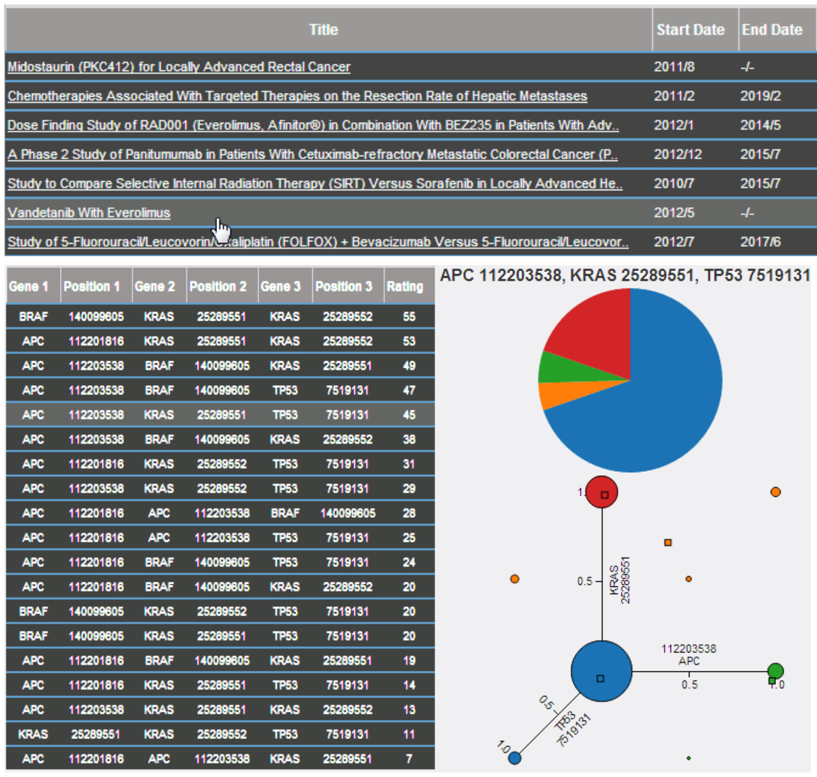
### B. Lightweight Compression

Lightweight compression refers to a data storage representation, which consumes less space than its original pendant [11]. A columnar storage layout, as used in IMDBs, supports lightweight compression techniques, such as run-length encoding, dictionary encoding, and difference encoding [13]. Typically, values of a database attribute are within a very small subset of the attribute's domain, e.g., `male` and `female` for the gender type. Lightweight compression maps all unique values to a uniform format, e.g., `male=1` and `female=2`.

### C. Partitioning

We distinguish between vertical and horizontal partitioning [14]. The former refers to the arrangement of database columns. It is achieved by splitting columns of one database table in multiple column sets wile each set can be distributed on individual servers [15]. The latter addresses long database tables and their division into smaller chunks of data. Splitting data into equally long horizontal partitions supports parallel search operations and improves scalability [11].

The identification of CpG Islands (CGIs) is a concrete application example. CGIs are known to represent unstable chemical compounds. Its identification requires a full scan of the genome table to find cytosine and guanine bases stored as direct neighbors [16]. Applying a horizontal partition per chromosome for the genome table enables scanning of all chromosomes by individual threads in parallel.



| Title | Start Date | End Date | Min Age | Max Age | Rating |
|---|---|---|---|---|---|
| Midostaurin (PKC412) for Locally Advanced Rectal Cancer | 2011/8 | -/- | 18/0/0/0/0 | 0/0/0/0/0 | 3/4 |
| Chemotherapies Associated With Targeted Therapies on the Resection Rate of Hepatic Metastases | 2011/2 | 2019/2 | 18/0/0/0/0 | 0/0/0/0/0 | 3/4 |
| Dose Finding Study of RAD001 (Everolimus, Afinitor®) in Combination With BEZ235 in Patients With Adv.. | 2012/1 | 2014/5 | 18/0/0/0/0 | 0/0/0/0/0 | 3/4 |
| A Phase 2 Study of Panitumumab in Patients With Cetuximab-refractory Metastatic Colorectal Cancer (P.. | 2012/12 | 2015/7 | 18/0/0/0/0 | 0/0/0/0/0 | 3/4 |
| Study to Compare Selective Internal Radiation Therapy (SIRT) Versus Sorafenib in Locally Advanced He.. | 2010/7 | 2015/7 | 18/0/0/0/0 | 0/0/0/0/0 | 3/3 |
| Vandetanib With Everolimus | 2012/5 | -/- | 0/0/0/0/0 | 0/0/0/0/0 | 3/3 |
| Study of 5-Fluorouracil/Leucovorin/[illegible]aliplatin (FOLFOX) + Bevacizumab Versus 5-Fluorouracil/Leucovor.. | 2012/7 | 2017/6 | 18/0/0/0/0 | 70/0/0/0/0 | 3/3 |

| Gene 1 | Position 1 | Gene 2 | Position 2 | Gene 3 | Position 3 | Rating |
|---|---|---|---|---|---|---|
| BRAF | 140099605 | KRAS | 25289551 | KRAS | 25289552 | 55 |
| APC | 112201816 | KRAS | 25289551 | KRAS | 25289552 | 53 |
| APC | 112203538 | BRAF | 140099605 | KRAS | 25289551 | 49 |
| APC | 112203538 | BRAF | 140099605 | TP53 | 7519131 | 47 |
| APC | 112203538 | KRAS | 25289551 | TP53 | 7519131 | 45 |
| APC | 112203538 | BRAF | 140099605 | KRAS | 25289552 | 38 |
| APC | 112201816 | KRAS | 25289552 | TP53 | 7519131 | 31 |
| APC | 112203538 | KRAS | 25289552 | TP53 | 7519131 | 29 |
| APC | 112201816 | APC | 112203538 | BRAF | 140099605 | 28 |
| APC | 112201816 | APC | 112203538 | TP53 | 7519131 | 25 |
| APC | 112201816 | BRAF | 140099605 | TP53 | 7519131 | 24 |
| APC | 112201816 | BRAF | 140099605 | KRAS | 25289552 | 20 |
| BRAF | 140099605 | KRAS | 25289552 | TP53 | 7519131 | 20 |
| BRAF | 140099605 | KRAS | 25289551 | TP53 | 7519131 | 20 |
| APC | 112201816 | BRAF | 140099605 | KRAS | 25289551 | 19 |
| APC | 112201816 | KRAS | 25289551 | TP53 | 7519131 | 14 |
| APC | 112203538 | KRAS | 25289551 | KRAS | 25289552 | 13 |
| KRAS | 25289551 | KRAS | 25289552 | TP53 | 7519131 | 11 |
| APC | 112201816 | APC | 112203538 | KRAS | 25289551 | 7 |

Figure 3. Selected HIG cloud applications: The upper Fig. shows the ranked results of our clinical trials search. It includes the patient's history and extracts relevant criteria from trial free-text descriptions with the help of specific rules to identify relevant studies, e.g., clinical trial for advanced rectal cancer started in Aug. 2011. The left Fig. shows the results of an interactive analysis of a cohort of colon carcinoma patients using k-means clustering. It proposes relevant combinations of genomic loci, such as gene KRAS and on chromosome 12 at position 25,289,551, which are present in the majority of cohort members.

## IV. Architecture

Our HIG system architecture modeled in Fig. 1 combines data from various data sources, such as patient-specific data, genome data, and annotation data within a single system to enable flexibly real-time analysis and combination. In the following, its layers are described in further detail.

### A. *Application Layer*

The application layer consists of special purpose applications to answer medical and research questions. We provide an Application Programming Interface (API) that can be consumed by various kinds of applications, such as web browser applications or mobile applications. Fig. 1 depicts the data exchange via asynchronous Ajax calls and JavaScript Object Notation (JSON) [17], [18]. As a result, performing specific analyses is no longer limited to a specific location, e.g., the desktop computer of a clinician. Instead, all applications can be accessed via device connected to the Internet, e.g., laptop, mobile phone, or tablet computer. Thus, having access to relevant data at any time enhances the userÕs productivity. Selected HIG cloud applications are depicted in Fig. 3, which are described in the following. The end user can access these cloud applications via any Internet browser after registration.

*1) Cohort Analysis:* The HIG cohort analysis application enables researchers and clinicians to perform interactive clustering on the data stored in the IMDB, e.g., k-means and hierarchical clustering [19, Chap. 13]. Thus, they are able to verify hypotheses by combining patient and genome data in real-time. Therefore, they use patient-specific and genome data loaded into the in-memory database system and perform the interactive cohort analysis as part of the platform layer as depicted in Fig. 1.

*2) Clinical Trial Search::* Our HIG clinical trial search assists physicians in finding adequate clinical trials for their patients. It analyses patient data, such as age, gender, preconditions and existing mutations, and matches them with open clinical trials. The analysis is performed on top of more than 30k descriptions of searching clinical trials, which are ranked in real-time while the physician investigates the list of variants in the patient's genome [20].

### B. *Platform Layer*

The platform layer holds the complete process logic and consists of the IMDB system for enabling real-time analysis of genomic data. We developed specific extensions for the IMDB system to support genome data processing and its analysis. In the following, selected extensions and their integration in the IMDB system are described in more detail.

*1) Scheduling of Data Processing:* We extended the IMDB by a worker framework, which executes tasks asynchronously, e.g., alignment of chunks of genome data. It consists of a task scheduler instance and a number of workers controlling dedicated computing resources, e.g., individual computing nodes. Workers retrieve tasks and parameters by the scheduler instance and perform specific tasks, such as workbench preparation, task execution, and maintenance of status information. Thus, all workers are connected to the IMDB to store status information about currently executed tasks.

Furthermore, the scheduler supervises the responsiveness of individual compute resources. If a predefined response behavior is no longer guaranteed, e.g., due to an overloaded compute node or a crashed worker process, workers are marked as unresponsive. As a result, work-in-progress tasks of the unresponsive worker are reassigned to a new worker and this worker is scheduled for a restart.

*2) Updater Framework:* We consider the use of latest international research results as enabler for evidence-based therapy decision [21]. The updater framework is the basis for combining international research results. It periodically checks all registered Internet sources, such as public FTP servers or web sites, for updated and newly added versions of annotations, e.g., database exports as dumps or characteristic file formats, such as CSV, TSV, and VCF. If the online version is newer than the local available version, the new data are automatically downloaded and imported in the IMDB to extend the knowledge base.

The import of new versions of research databases is performed as a background job without affecting the system's operation. We import new data without any data transformations in advance. Thus, they are available for real-time analysis without any delay [22], [23]. For example, the following selected research databases are checked regularly by our updater framework: National Center for Biotechnology Information (NCBI), Sanger's catalogue of somatic mutations in cancer, University of California, Santa Cruz (UCSC) [24], [25], [26].

### *C. Data Layer*

The data layer holds all required data for performing processing and analyzing of genomic data. The data can be distinguished in the two categories: master data, and transactional data [27]. For example, human reference genomes and annotation data are referred to as master data, whereas patient-specific NGS data and Electronic Medical Records (EMR) are referred to as transactional data [28], [29]. Its analysis is the basis for gathering specific insights, e.g., individual genetic dispositions, and for leveraging personalized treatment decision in course of personalized medicine [3].

The actual step of analyzing the genetic data requires answering very specific questions. Thus, the application layer consists of specific applications to answer these questions. They make use of the platform layer to initialize the data processing.

TABLE I. CONFIGURATION OF BWA BENCHMARKS.

| Experiment | Split Size | Primary Storage |
|---|---|---|
| 1 | 1 | File System |
| 2 | 1 | In-Memory Database |
| 3 | 25 | File System |
| 4 | 25 | In-Memory Database |

## V. Benchmarks

All benchmarks were performed on a computing cluster consisting of 25 identical computing nodes. Each of the nodes was equipped with four Intel Xeon E7-4870 CPUs running at 2.40 GHz clock speed, 30 MB Intel Smart cache[30], interconnected by 6.4 GT/s Quick Path Interconnect (QPI), and 1 TB of main memory capacity. Each CPU consisted of 10 physical cores and 20 threads running a 64-bit instruction set. All computing nodes were equipped with Intel 520 series Solid State Drives (SSDs) of 480 GB capacity combined using a hardware raid for local file operations [31]. The average throughput rate of the local SSDs was measured with 7.6 GB/s cached reads and 1.4 GB/s buffered disk reads. All nodes were interconnected via a Network File System (NFS) using dedicated 10 Gb/s Ethernet links and switches to share data between nodes.

Instead of using generated test data, we only used real NGS data, i.e. FASTQ files, from the 1,000 genome project for individual measurements [32]. We used the FASTQ file of patient HG00251 for our benchmarks. It consumed 160 GB of disk space, consists of approx. 63 Gbp, approx. 695 M reads with 91 bp individual read length forming approx. 20x coverage.

The aim of all conducted benchmarks was to minimize the overall execution time for a single GDPP run, i.e. to use the maximum available computing power to achieve best parallelization. In the following, we share insights about our benchmarks performed on our HIG system.

### *A. Performance Key Indicators*

We investigated the following impact factors to controlling the overall pipeline execution time:

1) **Integration:** We implemented GDPPs based on existing alignment and variant calling tools in our system architecture without any modification of the established tools,
2) **Adaption:** We adapted existing GDPPs to use the in-memory database as primary storage for data processing, and
3) **Optimization:** We optimized the number of involved distributed computing nodes (`split size`).

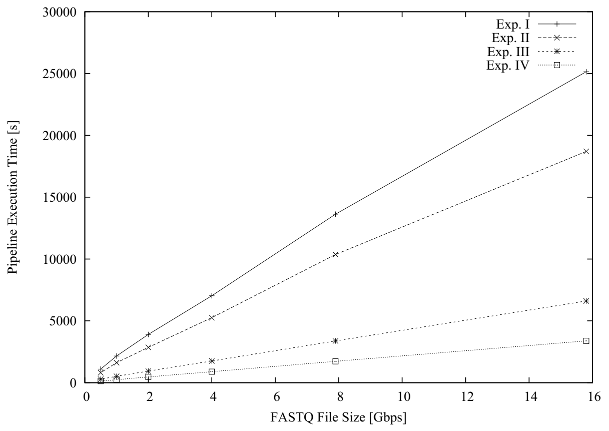


Figure 4. Comparison of benchmark results. Exp. 2 shows relative improvements of 24-27 % on single computing node compared to Exp. 1. Exp. 3 and 4 show relative improvements up 76 % and 89 % resp. on 25 computing nodes compared to Exp. 1 (Exp. = Experiment).

TABLE II. COMPARISON OF OVERALL PIPELINE EXECUTION TIMES IN HIG. $T_X$=EXECUTION TIME FOR EXP. X, $A_X$=RELATIVE ADVANCE OF EXECUTION TIME FOR EXP. X COMPARED TO EXP. 1 AS $A_X = \frac{T_1 - T_X}{T_1}$.

| Size $[Gbp]$ | 0.5 | 1.0 | 2.0 | 4.0 | 7.9 | 15.8 |
|---|---|---|---|---|---|---|
| $t_1[s]$ | 1,100 | 2,159 | 3,900 | 7,029 | 13,626 | 25,147 |
| $t_2[s]$ | 808 | 1,622 | 2,860 | 5,259 | 10,364 | 18,707 |
| $t_3[s]$ | 283 | 520 | 943 | 1,761 | 3,377 | 6,609 |
| $t_4[s]$ | 130 | 245 | 470 | 893 | 1,733 | 3,387 |
| $A_2[\%]$ | 27 | 25 | 27 | 25 | 24 | 26 |
| $A_3[\%]$ | 74 | 76 | 76 | 75 | 75 | 74 |
| $A_4[\%]$ | 88 | 89 | 88 | 87 | 87 | 87 |

### B. Results

Our benchmarks were performed on a GDPP integrating Burrows Wheeler Aligner (BWA) version 0.6.2 as selected alignment algorithm [33]. We designed our benchmarks to compare the impact of the incorporated storage and the level of parallelization on the overall execution time as outlined in Tab. I.

Exp. 1 and 2 used a file system as primary storage while an IMDB was used for Exp. 2 and 4. Exp. 1 and 2 were executed on a single computing node, while Exp. 3 and 4 were executed on 25 computing nodes to evaluate the impact of a fully parallelized execution environment.

The overall pipeline execution for varying FASTQ file sizes was measured. BWA was configured to use 80 threads, which relates to the maximum available hardware resources of our benchmark systems.

Exp. 1 and 2 describe the overall pipeline execution time on a single computing node as shown in Tab. II. It depicts that the use of the in-memory database as primary storage for intermediate results is beneficial for all selected file sizes. This pipeline optimization results in a reduction of runtime by at least 25 percent in average.

Exp. 3 and 4 as shown in Tab. II document the impact of the parameter `split size`, i.e. the number of distributed computing nodes used for parallel execution. Parallel execution of selected pipeline steps reduces the overall execution time by at least 74 percent in average for BWA. Additional improvement can be achieved by using the in-memory database as primary storage. Thus, the overall execution time can by reduced by at least 87 percent in average.

## VI. Evaluation and Discussion

Our conducted benchmarks verify two hypotheses. Firstly, the usage of the IMDB as primary storage system improves the overall execution time of established alignment algorithms, such as BWA, integrated in our HIG system as depicted in Fig. 4. Secondly, our HIG system supports the parallel execution of intermediate process steps across multiple computing nodes, which results in an additional performance improvement compared to the execution on a single computing node.

We observed the best relative improvement with the adapted pipeline using the IMDB as primary data storage with at least 74 percent on single computing node and up to 89 percent on 25 computing nodes. It shows that the overall pipeline execution time correlates to the number of base pairs contained in the FASTQ file in a very constant and linear way. However, the improvement of using 25 nodes is still below our expectation of a factor 25 since we also use traditional tools in the GDPP, which partially operate in a single threaded way.

The scaling factors 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 HIG system for varying input file size. It helps us to predict execution times and helps to supervise the correct system functionality, e.g., to detect broken computing resource as outlined in Sect. IV.

Furthermore, our results stress the benefits of using an IMDB for operating on intermediate results of the pipeline execution. The pipeline optimized for the IMDB no longer uses individual tools operating on files for specific process steps, such as sorting, merging, and indexing. In contrast, these operations are directly performed as an integral operation of the IMDB without the need to create intermediate files in the file system any longer. We integrated existing alignment and variant calling tools into our HIG architecture without modifying their code. Thus, the speed-up documented in our benchmarks is moderate and mainly achieved by replacing selected file-based by optimized in-memory database operations.

## VII. Conclusion and Outlook

In our contribution, we shared details about our HIG system enabling genome data processing on an IMDB. We outlined the applicability of this technology for processing of genome data to enable real-time analysis of genome data. Furthermore, we shared insights in specific design decision for our IT architecture, such as scheduler and updater framework. The presented benchmark results proved that our HIG system improves overall pipeline execution time by at least one fourth on a single computing node and up to 89 percent involving our computing cluster with 25 nodes. The performance improvements are achieved by substituting selected disk-based operations, such as sorting, merging, and indexing, by native in-memory database operations.

Our future research work focuses on optimizing alignment and variant calling algorithms executing them directly within the IMDB. As a result, the amount of media breaks and incorporated data transfers would be reduced further.

## References

[1] F. S. Collins, A. Patrinos, E. Jordan, A. Chakravarti, R. Gesteland, and L. Walters, "New Goals for the U.S. Human Genome Project," Science, vol. 282, no. 5389, 1998, pp. 682–689.

[2] W. J. Ansorge, "Next-generation DNA Sequencing Techniques," New Biotechn, vol. 25, no. 4, 2009, pp. 195–203.

[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, Real-time Security Extensions for EPCglobal Networks: Case Study for the Pharmaceutical Industry. Springer, 2013.

[6] A. Knöpfel, B. Grone, and P. Tabeling, Fundamental Modeling Concepts: Effective Communication of IT Systems. John Wiley & Sons, 2006.

[7] S. Pabinger et al., "A Survey of Tools for Variant Analysis of Next-generation Genome Sequencing Data," Brief Bioinform, Jan. 2013.

[8] S. Wandelt, A. Rheinländer, M. Bux, L. Thalheim, B. Haldemann, and U. Leser, "Data Management Challenges in Next Generation Sequencing," Datenbank-Spektrum, vol. 12, no. 3, 2012, pp. 161–171.

[9] National Human Genome Research Institute, "DNA Sequencing Costs," http://www.genome.gov/sequencingcosts/ [retrieved: Sep 20, 2013], Apr 2013.

[10] J. C. McCallum, "Memory Prices (1957-2013)," http://www.jcmit.com/memoryprice.htm [retrieved: Sep 20, 2013], Feb 2013.

[11] H. Plattner, A Course in In-Memory Data Management: The Inner Mechanics of In-Memory Databases, 1st ed. Springer, 2013.

[12] M.-P. Schapranow, "Transaction Processing 2.0," Master's thesis, Hasso Plattner Institute, 2008.

[13] P. Svensson, "The Evolution of Vertical Database Architectures – A Historical Review," in Proceedings of the 20th Int'l Conf on Scientific and Statistical Database Management. Springer-Verlag, 2008, pp. 3–5.

[14] S. S. Lightstone, T. J. Teorey, and T. Nadeau, Physical Database Design: The Database Professional's Guide to Exploiting Indexes, Views, Storage, and more. Morgan Kaufmann, 2007.

[15] J. M. Hellerstein, M. Stonebraker, and J. Hamilton, Architecture of a Database System, Foundation and Trends in Databases. now Publishers, 2007, vol. 1.

[16] M. Gardiner-Garden and M. Frommer, "CpG islands in vertebrate genomes," Mol Biol, vol. 196, no. 2, July 1987, pp. 261–282.

[17] A. T. Holdener, AJAX: The Definitive Guide, 1st ed. O'Reilly, 2008.

[18] D. Crockford, "RFC4627: The application/json Media Type for JavaScript Object Notation (JSON)," http://www.ietf.org/rfc/rfc4627.txt [retrieved: Sep 20, 2013], July 2006.

[19] S. Krawetz, Bioinformatics for Systems Biology. Humana Press, 2009.

[20] U.S. National Institutes of Health, "Clinicaltrials.gov," http://www.clinicaltrials.gov/ [retrieved: Sep 20, 2013], 2013.

[21] M.-P. Schapranow, H. Plattner, and C. Meinel, "Applied In-Memory Technology for High-Throughput Genome Data Processing and Real-time Analysis," in Upgrade Devices in Tele Medicine, 2013, pp. 35–42.

[22] A. Bog, K. Sachs, and H. Plattner, "Interactive Performance Monitoring of a Composite OLTP and OLAP Workload," in Proceedings of the Int'l Conf on Mgmt of Data 2012. Scottsdale, AZ, USA: ACM, 2012, pp. 645–648.

[23] F. Färber, S. K. Cha, J. Primsch, C. Bornhövd, S. Sigg, and W. Lehner, "SAP HANA database: Data Management for Modern Business Applications," SIGMOD Rec., vol. 40, no. 4, Jan. 2012, pp. 45–51.

[24] National Center for Biotechnology Information, "All resources," http://www.ncbi.nlm.nih.gov/guide/all/ [retrieved: Sep 20, 2013].

[25] S. A. Forbes et al., "The Catalogue of Somatic Mutations in Cancer: A Resource to Investigate Acquired Mutations in Human Cancer." Nucl Acids Res, vol. 38, 2010.

[26] L. R. Meyer et al., "The UCSC Genome Browser Database: Extensions and Updates 2013," Nucl Acids Res, 2012.

[27] T. K. Das and M. R. Mishra, "A Study on Challenges and Opportunities in Master Data Management," Int'l Journal of Database Mgmt Syst, vol. 3, no. 2, May 2011.

[28] The Genome Reference Consortium, "Genome Assemblies," http://www.ncbi.nlm.nih.gov/projects/genome/assembly/grc/data.shtml [retrieved: Sep 20, 2013].

[29] A. Rector, W. Nolan, and S. Kay, "Foundations for an Electronic Medical Record," Methods of Information in Medicine, 1991, pp. 179–186.

[30] Intel Corporation, "Intel Product Quick Reference Matrix," http://cache-www.intel.com/cd/00/00/47/64/476434_476434.pdf [retrieved: Sep 20, 2013], Apr 2011.

[31] ——, "Intel Solid-State Drive 520 Series Product Specification," http://www.intel.com/content/dam/www/public/us/en/documents/product-specifications/ssd-520-specification.pdf [retrieved: Sep 20, 2013], Feb 2012.

[32] The 1000 Genomes Project Cons., "A Map of Human Genome Variation from Population-scale Sequencing," Nature, vol. 467, no. 7319, Oct. 2010, pp. 1061—1073.

[33] H. Li and R. Durbin, "Fast and Accurate Short Read Alignment with Burrows-Wheeler Transformation," Bioinformatics, vol. 25, 2009, pp. 1754–1760.