A feedback guided interface for elastic computing

Sebastian Schönherr[1,2,*], Lukas Forer[1,2,*], Hansi Weißensteiner[1,2], Florian Kronenberg[2], Günther Specht[1], Anita Kloss-Brandstätter[2]

[*] contributed equally

[1]Databases and Information Systems
Institute of Computer Science
University of Innsbruck, Austria
sebastian.schoenherr@uibk.ac.at

[2]Division of Genetic Epidemiology
Department of Medical Genetics, Molecular and Clinical Pharmacology
Innsbruck Medical University, Austria
lukas.forer@i-med.ac.at

ABSTRACT

Computer Science plays an important role in today's Genetics. New sequencing methods produce an enormous amount of data, pushing genetic laboratories to storage and computational limits. New approaches are needed to eliminate these shortcomings and provide possibilities to reproduce current solutions and algorithms in the area of Bioinformatics. In this paper a system is proposed which simplifies the access to computational resources and associated computational models of cluster architectures, assists end users in executing and monitoring developed algorithms via a web interface and provides an interface to add future developments or any kind of programs. We demonstrate on existing algorithms how an integretation can be done with little effort, making it especially useful for the evaluation and simplified usage of current algorithms.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Distributed System, Experimentation, Application

Keywords

Bioinformatics, Hadoop, MapReduce, Cloud computing

23^{rd} GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), 31.05.2011 - 03.06.2011, Obergurgl, Austria.

1. INTRODUCTION

In recent years Computer Science became an essential part in the field of Genetics. Especially through the advent of Next Generation Sequencing (NGS), whereby a human genome (3 billion base pairs/chromosome set) can be sequenced in acceptable time, the amount of data is growing significantly, exceeding all known dimensions in Genetics. Figure 1 shows a comparison between the reducing DNA sequencing costs and Moore's law. Moore's law is used as a reference to show that computer hardware can currently not keeping pace with the progress in DNA sequencing. Furthermore, the amount of complete sequenced individuals is growing exponentially from year to year [11], making new models necessary. For instance, to store the data of *one* complete human DNA (Deoxyribonucleic acid) in raw format with 30-times coverage, 30 terabytes of data is produced.

In the area of *Copy Number Variations*, a possible cause for many complex genetic disorders, high throughput algorithms are needed to process and analyze several hundred gigabytes of raw input data [16] [6], yielding to a wall time of up to one week for a typical study size [18]. This remarkable increase of data and time causes genetic departments to consider new ways of importing and storing data as well as improving performance of current algorithms.

Cluster architectures in connection with associated models have the potential to solve this issue, but especially for small departments often gainless and unaffordable. Using clusters on demand, also referred to *Infrastructure as a Service* (IaaS), builds therefore a good opportunity to circle these issues. To capitalize the full potential of IaaS, a combination with distribution models like MapReduce [5] is for specific applications both possible and obvious. Several isolated applications [9], [10], [14] already exist using a distributed approach for storing data and processing algorithms. But since no general system is given to execute those solutions, an evaluation and reproducibility is often not feasible. Scientists need to setup a cluster on their own or using a provided remote cluster architecture to evaluate a published algorithm, being both time wasting and insecure for sensitive data.

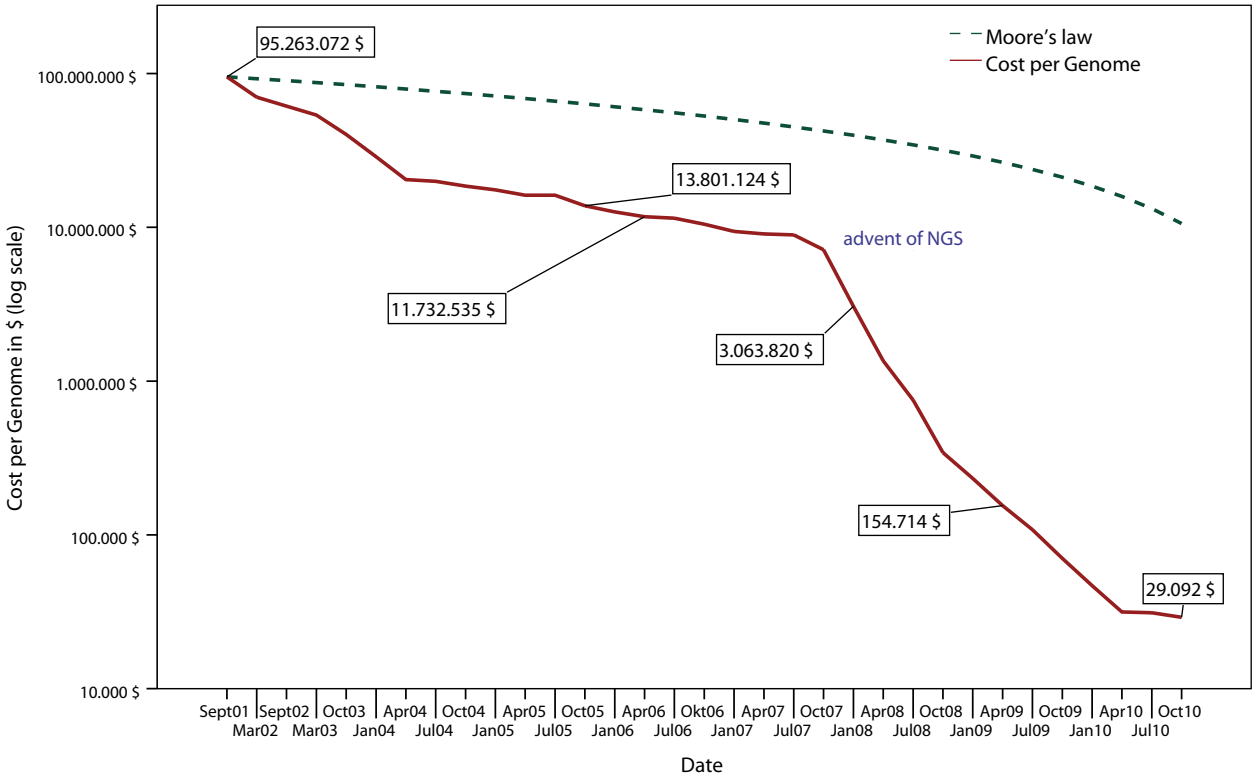

Figure 1: Comparision of DNA sequencing cost with Moore's law; Data from [17]

In this paper we present the idea to build an integrated system for scientists in the area of Bioinformatics to (1) get access to distributed cluster architectures and execute existing algorithms, (2) build maintainable and reproducible workflows and (3) provide an interface to add future developments or any kind of programs to the system without detailed IT knowledge. The reminder of this paper is structured as follows: Section 2 gives an overview of the related work. In section 3 the architecture of our suggested system is explained in more detail with potential case studies in section 4. Section 5 shows necessary future work and the paper ends with a conclusion in section 6.

2. RELATED WORK

Cluster solutions guided by a web-interface to execute distributed algorithms like Myrna [9], CrossBow [10] or CloudBurst [13] already exist. Unfortunately, the user must login to the Amazon Web Services (AWS) console to monitor the progress of executed jobs or to shutdown the cluster after execution. Additionaly, a data storage in S3 buckets is often required and a custom web interface needs to be implemented for every single approach.
Galaxy [7] is a software system which facilitates the creation, execution and maintainability of pipelines in a fast and user friendly way. The platform itself executes the scripts and the user has the possibility to monitor the progress. Galaxy's extension CloudMan [1] provides the possibility to install and execute Galaxy on Amazon EC2 (Elastic Compute Cloud). However, the user needs to start the master node manually by using the AWS console and Galaxy does not provide a native support of Hadoop programs, executes modules step by step and distributes only whole jobs among the cluster.

3. ARCHITECTURE

A modular architecture is suggested in Figure 2, separating the process of instantiate and set up a cluster (*Cloudgene*) from the process of monitor and run a program (*EMI*). Based on open source frameworks like Apache Hadoop [2] and Apache Whirr [3], we implemented a prototype to verify our approach. The user utilizes Cloudgene to set up a cluster architecture to his needs through XML configuration files. This allows adding new algorithms dynamically without digging into Cloudgene to deep. A fully operable and customized cluster is then provided, including all necessary user data. In a subsequent step EMI (Elastic MapReduce Interface) is launched on the master node of the cluster. EMI can be seen as an abstraction of the underlying system architecture from the end user, lies on top of the integrated programs and allows the user to communicate and interact with the cluster as well as receive feedback of currently executed workflows (see Figure 3). EMI can be disabled in case a program already includes an interface by its own, yielding to the most general approach to execute any kind of developed solution. Both parts can be operated separately via configuration files with clear defined input and output variables.

3.1 Cloudgene

Amazon provides with its EC2 the currently most developed service for public clouds in the area of IaaS. Cloudgene supports besides EC2 also Rackspace [12] to provide access to cluster infrastructure. As mentioned in the introduction

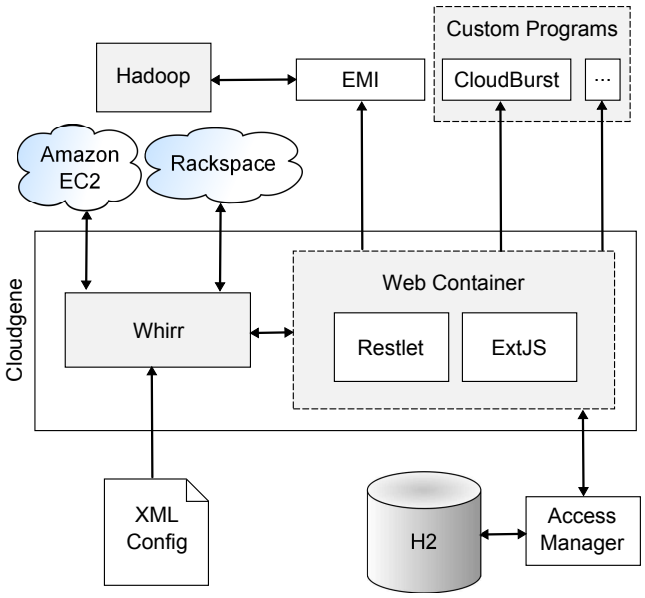

Figure 2: Architecture of the suggested system including Cloudgene and EMI

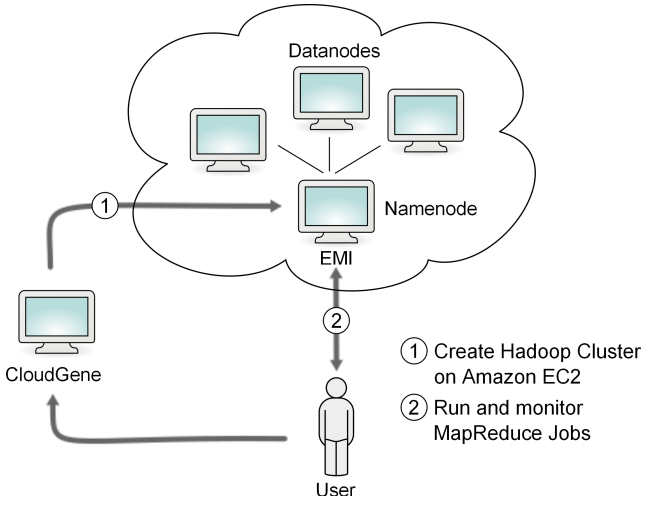

Figure 3: Workflow of the system including Cloudgene and EMI

a combination with MapReduce is useful: In this paradigm, the master node chops up data into chunks and distributes it over all active worker nodes (*map step*). Subsequently, the master node reassigns coherent map results to worker nodes (*sort and shuffle*) to calculate the final result (*reduce step*). For this project Apache Hadoop's implementation of MapReduce and its distributed file system (HDFS) are used. Using Whirr as a connector, Cloudgene is able to instance a full working EC2 or Rackspace cluster for end users with various defined properties and copies the necessary program data and configuration files to the cluster. Examples for defined variables could be the desired image, amount and kind of instances, HDFS options, MapReduce properties and the user's SSH public key. Amazon already provides several predefined images for all sorts of use cases, which can be be used with Cloudgene (e.g. http://www.cloudbiolinux.com). Cloudgene takes over the customization of predefined images and installs services like MapReduce, in our case included in Cloudera's distribution of Apache Hadoop [4]. The cluster configuration is defined in an XML-based file format, including all necessary information for a successful cluster boot. Cloudgene routinely checks if new configurations are added and offers the possibility to execute newly defined programs. Since EC2 is using a pay-per-use model, end users must provide their Amazon Access ID and Secret Key, which is transferred via Cloudgene to Amazon in a secure way. Alternatively, Cloudgene can also be launched on every machine having Java installed, eliminating the transfer via our server. Cloudgene solves one important issue and gives genetic departments access to computational power and storage. A still unresolved problem is the lack of a graphical user interface to control jobs deriving from command line based applications. Especially the need of putting enormous amount of local data into HDFS has to be considered. To overcome these shortcomings, a user interface (EMI) was designed.

3.2 Efficient MapReduce Interface (EMI)

Running Hadoop MapReduce programs on a cluster requires the execution of several non-trivial steps: First, the user must upload all input data to the master node, copy the data into the proprietary HDFS, run the Hadoop MapReduce job, export the results from the filesystem and finally download them to the local workstation. For researchers without expertise in Computer Science these tasks turns out to be very challenging. For this purpose we developed EMI which facilitates the execution, monitoring and evaluation of MapReduce jobs. A web interface, which runs on the master node of the cluster, enables the execution of jobs through well-structured wizards and setting all required parameters step by step. As several studies have shown, reproducibility of data analysis is one of the greatest problems in biomedical publications [15]. For this purpose the execution of a MapReduce job with its parameters and input data is logged, thus a fast comparison of experiments with different settings is possible. Moreover, the user always has the full control over an execution of each job and can monitor its current progress and status. All running jobs are listed whereby the progress of the map and reduce phase are displayed separately. Since using resources from Amazon costs money, EMI informs the user about the uptime of the cluster and the number of rented instances (Figure 4).
The modular architecture enables a fast integration of any Hadoop job which could be normally executed through the

command line. A simple and clear XML configuration file describes the input and output parameters of the program and contains other relevant information that are necessary to start the job (see Section 4). In addition to this file, a zip archive file exists which contains all software relevant data (e.g. jar file, meta data, configuration files). With those files, EMI automatically generates a web interface in which the possibility to set each defined parameter through wizards and to run the defined job by a single click is provided. As mentioned earlier, all input data must be put into the robust and fault-tolerant HDFS. As this process is very time-intensive an error prone, EMI supports the user by providing a wizard which enables the import of data from different sources (FTP, HTTP, Amazon S3 buckets or local file uploads). In addition, files defined as output parameters can be exported and downloaded as a zip archive or can be uploaded to Amazon S3 or FTP servers. EMI supports a multi-user mode whereby all data by a certain user are password protected and executed jobs are scheduled through a queue system. Overall, EMI is fully independent from Cloudgene and can be installed on a local Hadoop cluster too.

4. CASE STUDIES

In this section we explain how new programs can be integrated into Cloudgene and EMI. Based on two different biomedical software solutions we demonstrate the diversity and simplicity of our approach.

4.1 CloudBurst

CloudBurst is a parallel read-mapping algorithm to map NGS data to the human genome and other reference genomes [13]. It is implemented as a MapReduce program using Hadoop and can be executed with the following command:

```
hadoop jar emi/cloudburst/CloudBurst.jar \
  reference_genome reads results 36 36 3 0 1 240 \
  48 24 24 128 16
```

In order to execute CloudBurst we create a configuration file for Cloudgene which starts a Hadoop cluster on Amazon EC2 with a standard Ubuntu Linux with open Hadoop ports 50030 and 50070. The corresponding XML has the following structure:

```
<cloudgene>
  <name>CloudBurst</name>
  <options>
    <option name="provider" value="amazon-aws"/>
    <option name="image" value="default"/>
    <option name="service" value="hadoop"/>
    <option name="emi" value="true"/>
    <option name="ports" value="50030 50070"/>
  </options>
</cloudgene>
```

As CloudBurst has no graphical user interface, we install EMI on the Amazon EC2 cluster and use it for user interactions. For this purpose the command above with its arguments must be translated into the following configuration file:

```
<emi>
  <program>
    <name>CloudBurst</name>
    <command>
       hadoop jar emi/cloudburst/CloudBurst.jar \
          $input1 $input2 $output1 36 36 3 0 1 240 \
          48 24 24 128 16
    </command>
    <input>
      <param id="1" type="hdfs">
        <name>Reference Genome</name>
        <default>data/cloudburst/s_suis.br</default>
      </param>
    </input>
    <input>
      <param id="2" type="hdfs">
        <name>Reads</name>
        <default>data/cloudburst/100k.br</default>
      </param>
    </input>
    <output>
      <param id="1" type="hdfs" merge="true">
        <name>Results</name>
        <default>data/cloudburst/results</default>
      </param>
    </output>
  </program>
</emi>
```

After the XML file is uploaded to the Cloudgene server, the user starts a web browser to (1) login to Cloudgene, (2) start up a cluster preconfigured with CloudBurst and (3) run and monitor jobs with EMI (Figure 4).

Compared to a standard manual approach, this eliminates error-prone and time-consuming tasks such as (1) setting up a cluster and connecting via the command line onto the master node, (2) uploading and importing data into HDFS, (3) exporting final results from HDFS and downloading them and (4) executing and reproducing MapReduce jobs with different configurations via a web interface. This shows, that an easy integration can be done using a simple XML configuration, supporting and guiding researchers as far as possible.

4.2 HaploGrep

HaploGrep is a reliable algorithm implemented in a web application to determine the haplogroup affiliation of thousands of mitochondrial DNA (mtDNA) profiles genotyped for the entire mtDNA or any part of it [8]. As HaploGrep provides its own web interface we do not need to install EMI. Since it does not use the Hadoop service either, we note this option in the configuration as well. HaploGrep listens on the ports 80 (http) and 443 (https), therefore this ports are marked as open. The configuration file for Cloudgene with all requirements looks as follows:

```
<cloudgene>
  <name>Haplogrep</name>
  <options>
    <option name="provider" value="amazon-aws"/>
    <option name="image" value="default"/>
    <option name="service" value="none"/>
    <option name="emi" value="false"/>
    <option name="ports" value="80 443"/>
```

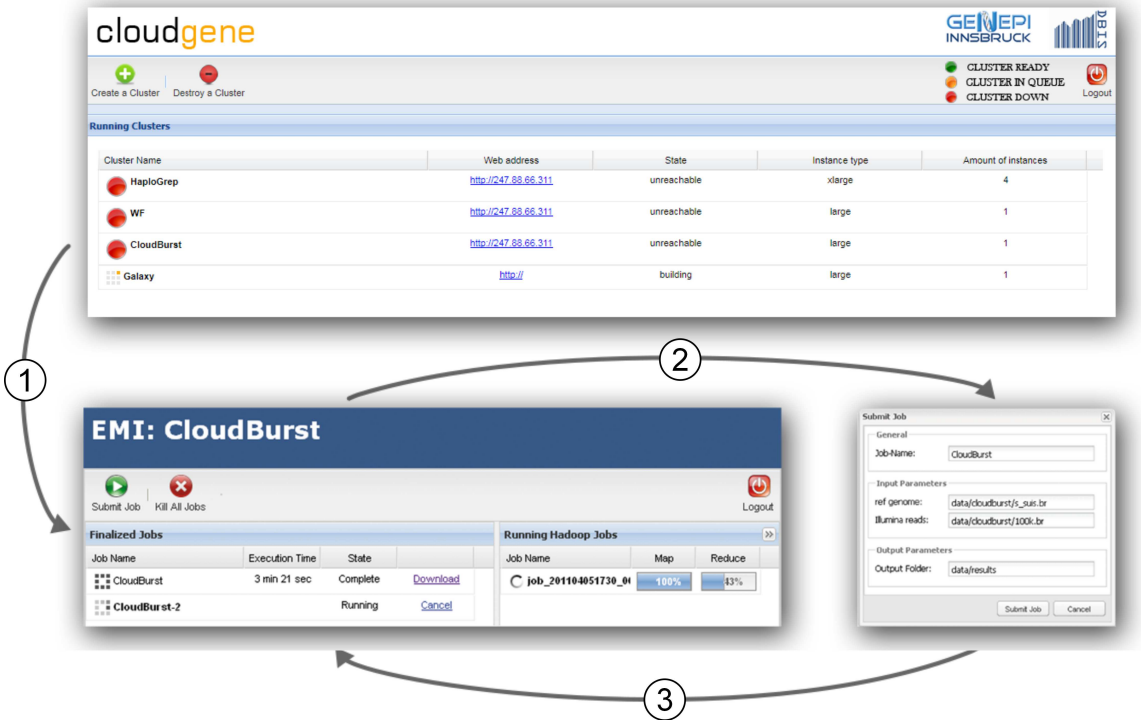

Figure 4: Workflow based on CloudBurst

```
  </options>
</cloudgene>
```

After the cluster setup is finalized, Cloudgene returns a web address which points to the installed instance of HaploGrep.

5. FUTURE WORK

One of the biggest advantages of IaaS is the changable amount of needed datanodes on demand. Thus, the next version of Cloudgene is conceived to provide functions for adding and removing instances during runtime. Currently, clusters started with Cloudgene are not data persistent which yields to a data loss after a shutdown is fulfilled. For this purpose we plan to store all results on persistent Amazon EBS volumes. Furthermore, a simple user interface for Hadoop is not only useful for the end user but also for developers. It supports them during the whole prototyping and testing process of novel MapReduce algorithms by highlighting performance bottlenecks. Thus, we plan to implement time measurements of the map, reduce and shuffle phase and to visualize them in an intuitive chart. Additionally, Hadoop plans in its next generation approach to support alternate programming paradigms to MapReduce, what is particularly important for applications (e.g. K-Means) where custom frameworks out-perform MapReduce by an order of magnitude.

6. CONCLUSION

We presented a software system for running and maintaining elastic computer clusters. Our approach combines the individual steps of setting up a cluster into a user-friendly system. Its modular architecture enables a fast integration of any Hadoop job which could be only executed through the command line. By hiding the low-level informatics, it is the ideal system for researchers without deeper knowledge in Computer Science. Moreover, our system is not constricted to the life sciences and can be used in nearly every application range. Overall, it is a first approach in order to narrow the gap between cloud-computing and usability.

7. ACKNOWLEDGMENTS

Sebastian Schönherr was supported by a scholarship from the University of Innsbruck (Doktoratsstipendium aus der Nachwuchsförderung, MIP10/2009/3). Hansi Weißensteiner was supported by a scholarship from the Autonomous Province of Bozen/Bolzano (South Tyrol). The project was supported by the Amazon Research Grant. We thank the Whirr Mailinglist especially Tom White and Andrei Savu for their assistance.

8. REFERENCES

[1] E. Afgan, D. Baker, N. Coraor, B. Chapman, A. Nekrutenko, and J. Taylor. Galaxy CloudMan: delivering cloud compute clusters. *BMC Bioinformatics*, 11 Suppl 12:S4, 2010.

[2] Apache Hadoop. http://hadoop.apache.org.

[3] Apache Whirr. http://incubator.apache.org/whirr/.

[4] Cloudera. http://www.cloudera.com/.

[5] J. Dean and S. Ghemawat. MapReduce: simplified data processing on large clusters. In *OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*, pages

10–10, Berkeley, CA, USA, 2004. USENIX Association.

[6] L. Forer, S. Schönherr, H. Weißensteiner, F. Haider, T. Kluckner, C. Gieger, H. E. Wichmann, G. Specht, F. Kronenberg, and A. Kloss-Brandstätter. CONAN: copy number variation analysis software for genome-wide association studies. *BMC Bioinformatics*, 11:318, 2010.

[7] J. Goecks, A. Nekrutenko, J. Taylor, E. Afgan, G. Ananda, D. Baker, D. Blankenberg, R. Chakrabarty, N. Coraor, J. Goecks, G. Von Kuster, R. Lazarus, K. Li, A. Nekrutenko, J. Taylor, and K. Vincent. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.*, 11:R86, 2010.

[8] A. Kloss-Brandstättter, D. Pacher, S. Schönherr, H. Weißensteiner, R. Binna, G. Specht, and F. Kronenberg. HaploGrep: a fast and reliable algorithm for automatic classification of mitochondrial DNA haplogroups. *Hum. Mutat.*, 32:25–32, Jan 2011.

[9] B. Langmead, K. D. Hansen, and J. T. Leek. Cloud-scale RNA-sequencing differential expression analysis with Myrna. *Genome Biol.*, 11:R83, 2010.

[10] B. Langmead, M. C. Schatz, J. Lin, M. Pop, and S. L. Salzberg. Searching for SNPs with cloud computing. *Genome Biol.*, 10:R134, 2009.

[11] R. E. Mills et al. Mapping copy number variation by population-scale genome sequencing. *Nature*, 470:59–65, Feb 2011.

[12] Rackspace. http://www.rackspace.com.

[13] M. C. Schatz. CloudBurst: highly sensitive read mapping with MapReduce. *Bioinformatics*, 25:1363–1369, Jun 2009.

[14] M. C. Schatz. The missing graphical user interface for genomics. *Genome Biol.*, 11:128, 2010.

[15] L. Shi et al. The balance of reproducibility, sensitivity, and specificity of lists of differentially expressed genes in microarray studies. *BMC Bioinformatics*, 9 Suppl 9:S10, 2008.

[16] K. Wang, M. Li, D. Hadley, R. Liu, J. Glessner, S. F. A. Grant, H. Hakonarson, and M. Bucan. PennCNV: An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. *Genome Research*, 17(11):1665–1674, Nov. 2007.

[17] Wetterstrand, K. A. DNA Sequencing Costs: Data from the NHGRI Large-Scale Genome Sequencing Program Available: http://www.genome.gov/sequencingcosts; Accessed 04/11/11.

[18] H. E. Wichmann, C. Gieger, and T. Illig. KORA-gen–resource for population genetics, controls and a broad spectrum of disease phenotypes. *Gesundheitswesen*, 67 Suppl 1:26–30, Aug 2005.