NGS workflow Optimization using a Hybrid Cloud Infrastructure

Lorenzo Mossucca, Olivier Terzo, Klodiana Goga
Infrastructure and Systems for Advanced Computing (IS4AC)
Istituto Superiore Mario Boella (ISMB)
Torino, Italy
mossucca@ismb.it, terzo@ismb.it, goga@ismb.it

Andrea Acquaviva, Francesco Abate, Rosalba Provenzano
Department of Control and Computer Engineering
Politecnico di Torino
Torino, Italy
andrea.acquaviva@polito.it, francesco.abate@polito.it

***Abstract*—E-science applications involve great deal of data, to satisfy these processing requests, distributed computing paradigms, such as cluster, Grid, Virtual Grid, Cloud Computing, and Hybrid System are growing exponentially. Existing computing infrastructures, software system design, and use cases have to take into account the enormity in volume of requests, size of data and computing load. In Bioinformatics field, such as in Next Generation Sequencing technology, in order to have more accurate analysis, it increases the amount of data to process. A new protocol for sequencing the messenger RNA in a cell, known as RNA-Seq, produces millions of short sequence fragments in a single run. These fragments can be used to measure levels of gene expression and to identify novel splice variants of genes. The proposed solution allows to make the system scalable and flexible reducing elaboration time. The first aspect covers reverse engineering of a fast splice junction mapper for RNA-Seq reads called TopHat in order to make parallelizable tasks and the second aspect concerns the development of hybrid architecture integrating a Grid and a Virtual Grid Environment.**

***Keywords*-grid computing; cloud computing; virtual; next generation sequencing; hybrid architecture.**

I. INTRODUCTION

Next Generation Sequencing (NGS) technologies, also known as second generation, have revolutionized biology and genomic research with the ability to draw from a single experiment a larger amount of data sequence [1] than with the previous technology known as Sanger Sequencing. DNA sequencing includes several methods and technologies that are used to determine the order of the nucleotide bases (adenine, guanine, cytosine, and thymine)in a molecule of DNA. The main novelty introduced by the NGS platform is to obtain smaller fragments from the molecules of DNA/RNA, called read, which are sequenced in parallel thus reducing the processing time [2], [3]. Aberrant mutations in the RNA transcription, as chimeric transcripts, are on the base of various forms of disease and NGS proved to be extremely helpful in making the detection of these events more accurate and reliable. However, even if from the biological point of view NGS technology leads to new exciting perspectives spreading an incredible amount of data, it raised new challenges in the development of tools and computing infrastructures. A NGS machine generates millions of reads in a single run that must be successively elaborated and analyzed. TopHat is a program that aligns RNA-Seq reads to a genome in order to identify exon-exon splice junctions. It is built on the ultrafast short read mapping program Bowtie. TopHat finds splice junctions without a reference annotation. By first mapping RNA-Seq reads to the genome, TopHat identifies potential exons, since many RNA-Seq reads will contiguously align to the genome. Using this initial mapping, TopHat builds a database of possible splice junctions, and then maps the reads against this junction to confirm them. Out objective is to offer to biologist private infrastructures to conduct their research and to respond to the ever evolving needs of NGS users. Cloud Computing is rapidly emerging as an alternative platform for the computational and data needs of our community. Biologists are already using the Amazon Elastic Cloud Computing (EC2) infrastructure for their research. In some situations where such a sensitive data are involved, for privacy and data security issue, it is preferable to use a number of instances of a tailored Virtual Machine (VM) than submitting jobs to the own existing infrastructure. Grids appear mainly in high performance computing environments. In this context, several of off-the-shelf nodes can be linked together and work in parallel to solve problems, that, previously, could be addressed sequentially or by using supercomputers. Grid Computing is a technique developed to elaborate enormous amounts of data and enables large-scale resource sharing to solve problem by exploiting distributed scenarios. The main advantage of Grid is due to parallel computing, indeed if a problem can be split in smaller tasks, that can be executed independently, its solution calculation speed up considerably. The paper is organized as follows: In Section II, main issues of the NGS technology are discussed that led to the realization of this project. Section III explains biological background, software and tools used. Section IV, designed architecture is shown: grid and virtual environment and schedulers functionalities. Section V is related to the test performance. The last section draws the conclusion and directions for future works.

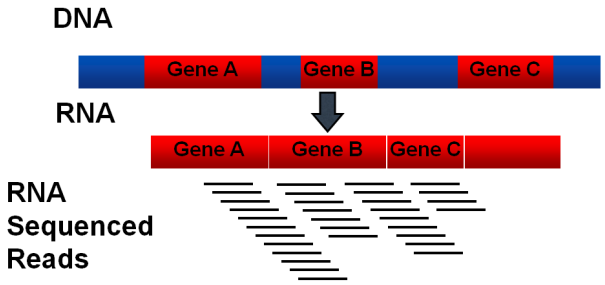

Figure 1. Alignment Phase.

II. Motivation

The amount of data produced with NGS technology is a positive factor that on a hand contributes to make studies more accurate and reliable identification of mutations in aberrant splicing events, fused genes, on the other hand, open new challenges in the development tools and infrastructure that are able to do the post-processing of data produced in a powerful and timely fashion [4]. A NGS data sample consists of millions of reads, and in a classic situation, with only one workstation available, the time needed to obtain the output increases significantly. In such a context, this computing infrastructure allows to improve overall system performance optimizing the use of resources and increasing the system scalability. The alignment is a process in which each mapping reference is made to read independently from the other reads, and this means that you can perform a parallel analysis of data. Even if the alignment is a very basic operation, due to the great number of data involved in the process, the computational effort in this phase is very high. This scenario recalls for the need of developing computing infrastructures presenting high performances CPU capability and memory availability [5].

III. Next Generations Sequencing

The aim of splicing tools is to track inside of the analyzed samples, the junctions between exons generated as a result of the process of splicing. This process allows that from the pre-mRNA introns are deleted and the remaining exons are joined together to form the mature RNA. The junction between exons spliced RNA is called splicing point, and thanks to the identification of these points it allows to figure out if events have occurred or whether alternative splicing events have occurred or whether we are in the presence of fused genes. There are several tools for splicing such as Supersplat, Splitseek and TopHat, in this section a brief overview for each tool is shown. Supersplat is a method for unbiased splice junction discovery through empirical RNA-seq data. Using a genomic reference and RNA-seq high throughput sequencing dataset, it empirically identifies potential splice junctions at a rate of approximately 114 million read per hour. The method depends upon the availability of previously known splice junctions on which to train the algorithm, and, when finding novel potential splice junctions, is biased toward those which are similar to its training data. In scoring novel potential splice junctions, the algorithm is biased toward canonical terminal dinucleotides, scoring those which conform to these biases higher than ones that do not. While, in general, these biases may prove to be correct, many potential splice junctions which do not conform to these rules threaten to remain unidentified. SplitSeek performs a number of analysis steps to predict the exon boundaries. First, all instances of split reads are found, and their genomic positions and nucleotide sequence are recorded. They comprise the initial set of candidates, and all resulting splice events will be found among these. However, many reads exist in which the junction is located in one of the anchors rather than in the gap. To identify such additional junction reads, it allows to scan all reads in which only one of the anchors was aligned. If such an anchor can be extended to the exact position as a previously identified candidate junction, and the sequence in the two reads aligns perfectly within the first five bases of the other exon, then the read is considered to confirm the junction. SplitSeek can find junction reads in which as few as five bases overlap with the other exon. In the final step, all identified junction reads are grouped, and user-defined cut-offs are applied to obtain a final set of exon boundaries. Because this method is unbiased, it will report all types of events in which a read must be split to match the reference genome, including small insertions and deletions.

A. TopHat algorithm

TopHat [6] is a fast splice junction mapper for RNA-Seq reads that aligns RNA-Seq reads to mammalian-sized genomes using the ultra high-throughput short read aligner Bowtie, and then analyzes the mapping results to identify splice junctions between exons. TopHat is a collaborative effort between the University of Maryland Center for Bioinformatics and Computational Biology and the University of California, Berkeley Departments of Mathematics and Molecular and Cell Biology. On the market there are several NGS platforms, Roche 454, Solid and Illumina/Solexa, which differ from each other for the amount of data sequenced to each experiment. In this case, TopHat receives as input reads produced by the Illumina Genome Analyzer, although users have been successful in using TopHat with reads from other technologies. The input samples consist of two files of about 37 million of reads each. The two files are FASTA formatted paired-end reads. Dealing with paired-end reads means that the reads are sequenced by the sequencing machine only on the end of the same DNA/RNA molecule, thus the sequence in the middle part is unknown. Each sequenced end of the same read is also referred as mate. It results in two distinct files: the first mate of the same reads and the opposite mate. TopHat finds junctions by mapping reads

to the reference in two phases. In the first phase, the pipeline maps all reads to the reference genome using Bowtie. All reads that do not map to the genome are set aside as initially unmapped reads. Bowtie reports, for each read, one or more alignment containing no more than a few mismatches in the 5'-most s bases of the read. The remaining portion of the read on the 3' end may have additional mismatches, provided that the Phred-quality-weighted Hamming distance is less than a specified threshold. TopHat allows Bowtie to report more than one alignment for a read, and suppresses all alignments for reads that have more than this number. This policy allows so called multireads from genes with multiple copies to be reported, but excludes alignments to low-complexity sequence, to which failed reads often align and then assembles the mapped reads. TopHat extracts the sequences for the resulting islands of contiguous sequence from the sparse consensus, inferring them to be putative exons. TopHat produces a compact consensus file containing called bases and the corresponding reference bases in order to generate the island sequences. TopHat uses the reference genome to call the base. Because most reads covering the ends of exons will also span splice junctions, the ends of exons in the pseudoconsensus will initially be covered by few reads, and as a result, an exons pseudoconsensus will likely be missing a small amount of sequence on each end. In order to capture this sequence along with donor and acceptor sites from flanking introns, TopHat includes a small amount of flanking sequence from the reference on both sides of each island. TopHat has a number of parameters and options, and their default values are tuned for processing mammalian RNA-Seq reads also it can be used for another class of organism.

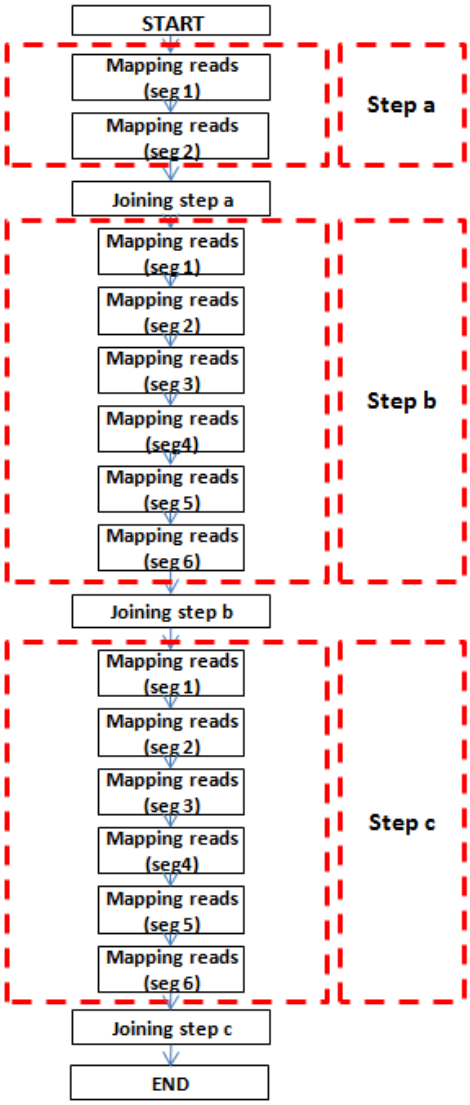

Figure 2. Sequential TopHat Workflow.

B. Alignment Tools: Bowtie

The short reads alignment is surely the most common operation in RNA-Seq data analysis [7]. The purpose of the alignment is to map each short read fragment onto a genome reference (see Figure 1). From the computational point of view, each short read consists of a sequence of four possible characters corresponding to the DNA bases and the sequence length depends on the sequencing machine adopted for the biological experiment [8]. The main novelty introduced by NGS technology is the capability of sequencing small DNA/RNA fragments in parallel, increasing the throughput and producing very short reads as output. However, this feature make the computational problem more challenging because of the higher amount of read produced and the accuracy in the mapping (shorter sequence length, higher probability of having multiple matches). For this reason many alignment tools specifically focussed on the alignment of short reads have been recently developed. In the present work, we are interested in characterizing the performances of alignment tools on real NGS data. Given our context, TopHat has been chosen, a wide diffused alignment program particularly aimed at aligning short reads. In order to detect the actual limitation of the alignment phase, we considered real NGS data coming from the analysis of Chronic Myeloid Leukemia. In our analysis flow, the HG19 assembly produced in 2007 is considered as reference genome the last human genome assembly produced to now. In order to increase the computational performances during the read mapping, Bowtie program creates an index of the provided human genome reference. This operation is particularly straightforward from the computational point of view, but it must be performed only one time for the human genome reference and it is independent on the mapping samples. The alignment phase itself is particularly suitable to be parallelized. In fact, each mapping operation is applied to each read independently on the other read mapping.

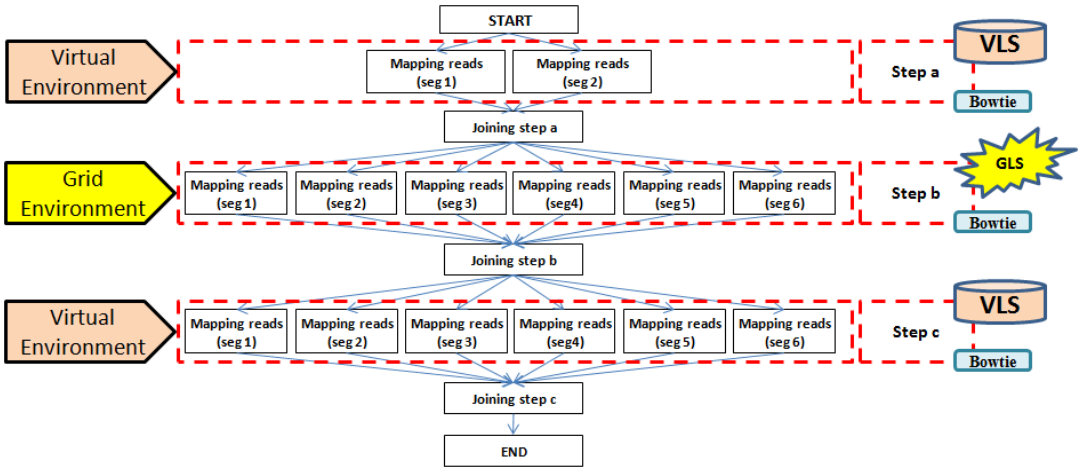

Figure 3. Distributed TopHat Workflow.

IV. Related work

Myrna [9] is a cloud computing tool for calculating differential gene expression in large RNASeq datasets. It allows to integrate short read alignment with interval calculations, normalization, aggregation and statistical modeling in a single computational pipeline. After alignment, Myrna calculates coverage for exons, genes, or coding regions and differential expression using either parametric or non-parametric permutation tests. The main advantage is the ability to rapidly test multiple plausible models for RNA-Seq differential expression. It has been suggested that this type of flexibility is necessary for computational applications to keep pace with the rapidly increasing number of reads in NGS data sets. Myrna allows to show that biological replicates reflect substantially increased variation compared to technical replicates in RNA-Seq and demonstrate that the commonly used Poisson model is not appropriate for biological replicates. It is designed to be run on the cloud using Amazon Elastic MapReduce, on any Hadoop cluster, or on a single computer. MapReduce is a programming model for processing large data sets implemented by Google. Usually, MapReduce is used to do distributed computing on clusters of computers. The model is inspired by the map and reduce functions commonly used in functional programming. MapReduce is a framework for processing parallel problems across huge datasets using a large number of node as cluster (Virtual Environment, Grid and Cloud Computing). Computational processing can occur on data stored either in a unstructured filesystem or in a structure database (also distributed DB). Algorithm is composed of two steps: Map step consists of divides task into smaller tasks, and distributes them to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes the smaller tasks, and passes the answer back to its master node. Reduce step: consists of collects the answers to all the sub-problems received from worker nodes and combines them in some way to form the output. MapReduce allows for distributed processing of the map and reduction operations. Main features is that each mapping operation must be independent of the others, so all maps can be performed in parallel. Hadoop is an open source software framework derived from Google's Map Reduce and Google File System for reliable, scalable, distributed computing of huge amounts of data. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures. Hadoop allows to split them into smaller sub-tasks, before reducing the results into one master calculation. This technique is not new, it has been conceived with the grid computing that has a new life with the coming of cloud computing. AWS (Amazon Web Services) offers a set of infrastructure and application services that allow the users to run virtually enterprise applications, big data projects, social games and mobile apps. Since 2009, Amazon launched Elastic Map Reduce which provides a hosted and scalable Hadoop service. Moreover Amazon offers other big data related services such as the Simple Queue Service for coordinating distributed computing and RDS a Managed Relational Database Service. Elastic Map Reduce (EMR) is the Amazon's implementation of Hadoop, it can be programmed in a usual way with tools like Pig and Hive. Instead of HDFS (Hadoop Distributed File System), as main data source, Amazon can be used as Simple Storage Service (Amazon S3) in order to get data in and out. But when using structured data S3 can be unwieldy, for this reason Amazon offers DynamoDB which is the No SQL storage solution to cover functionality of Hadoop HBase. Amazon

offers also EC2 (Elastic Compute Cloud) [11] which is a web service that provides resizable compute capacity in the cloud. It offers a web service through which users can boot an Amazon Machine Image (AMI) to create a virtual machine ”instance” and rent it. Amazon offers pay-per-use services, then resources are paid in a manner based on compute capacity of an instance, the size of storage requests (GB) and network traffic generated. Elastic IP addresses are static IP (IPv4) designed for dynamic cloud computing. An Elastic IP address belongs to the account and not to a virtual machine instance. It exists until it is explicitly released by the user. Amazon EC2 is based on the XEN para-virtualization technology and it is possible to sizes instances based on EC2 Compute Unit (ECU). One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor. Therefore the features of the virtual machine (such as the number of CPU cores, the processing power per core, memory, performance I/O, etc.) are fully user configurable and it is possible to choose from a series of configurations of the server. Even the operating system and the software that runs on the virtual server can be completely customized by the user.

V. METHODOLOGY

The proposed solution provides novelty on two aspects, firstly an optimization of the read mapping algorithm has been designed, in order to parallelize processes, secondly an implementation of a Hybrid architecture which consists of a Grid platform, composed of physical nodes, a Virtual platform, composed of virtual nodes set up on demand, and a scheduler devoted to integrate the two platforms.

A. Reverse Engineering

In a preliminary phase of reverse engineering, studying TopHat, blocks of transactions have been highlighted that were executed sequentially. We identified three main blocks, that can be executed independently:

a left and right mate mapped with HG19;
b segments mapped with HG19;
c segments mapped with segment juncs.

A feature of these three blocks is that they are performed by an external software, called Bowtie, as explained before. In steps (a) and (c), since the files involved in the development are significant, a common repository has been created that contains the temporary folders used by TopHat. Although, the use of a common repository slightly increases processing time, due to the SSH protocol connection, this time is lower than the time of transfer of the entire set of files to other machines. Instead the step (b) uses small files these can be performed on Grid environment, both physical and virtual, because the transfer times are lower. The only difference is that the input files are transferred to worker nodes through Globus Toolkit. These worker nodes re-send the output file to the node that requested execution when the process is terminated.

B. VirtualBio Infrastructure

The proposed architecture allows to manage RNA data, prepared by the version of TopHat in Grid, but not only, it could also handle other processing flows that use software and other tools, e.g., original version of TopHat or only Bowtie (see Figure 4). The architecture, called VirtualBio, is composed of three main components: a Master Node (MN), a part consists of the Physical Worker Nodes (PWN) that set the grid environment while a part consists of Virtual Worker Nodes (VWN) that set the virtualized environment. The MN is a physical machine with quite good hardware characteristics, is responsible for CA, contains the database, where all information about the nodes belonging to the infrastructure are stored, the node status, the flow of the various biological analysis that can be made in the system and system monitoring. Moreover, on it has been configured the common repository, using the Network File System (NFS). NFS is a protocol developed by Sun Microsystems in 1984 to allow computers to share files and folders over a network [13]. NFS is an open standard, defined in RFCs, and allows anyone to implement the protocol. Both environments are configured with the middleware Globus Toolkit [14], since it allows obtaining a reliable information technology infrastructure that enables the integrated, collaborative use of computers, networks, and databases. The Globus Toolkit is a collection of software components designed to support the development of applications for high performance distributed computing environments, or computational grids. OpenNebula.org [15] is an open-source project for building and managing virtualized enterprise data centers and cloud infrastructures. OpenNebula organizes existing storage, networking, virtualization, monitoring, and security platforms to enable the dynamic placement of multi-tier services. OpenNebula offers:

- The Image Repository System: it allows to set up and share images (e.g. operating systems or data) to be used in Virtual Machines easily.
- The Template Repository System: it allows to register Virtual Machine definitions in the system, to be instantiated later as Virtual Machine instances.
- Virtual Networking to interconnect Virtual Machines, they can be defined as fixed or ranged networks.
- As soon as a Template is instantiated to a Virtual Machine, can be performed a number of operations to control their lifecycle (migration, stop, resume, cancel, etc.). These operations are available both from the CLI and the Sunstone GUI.

Open Nebula corresponds most to the definition of a hybrid cloud. It integrates external resources in the cloud and does not introduce its own proprietary infrastructure. It has a

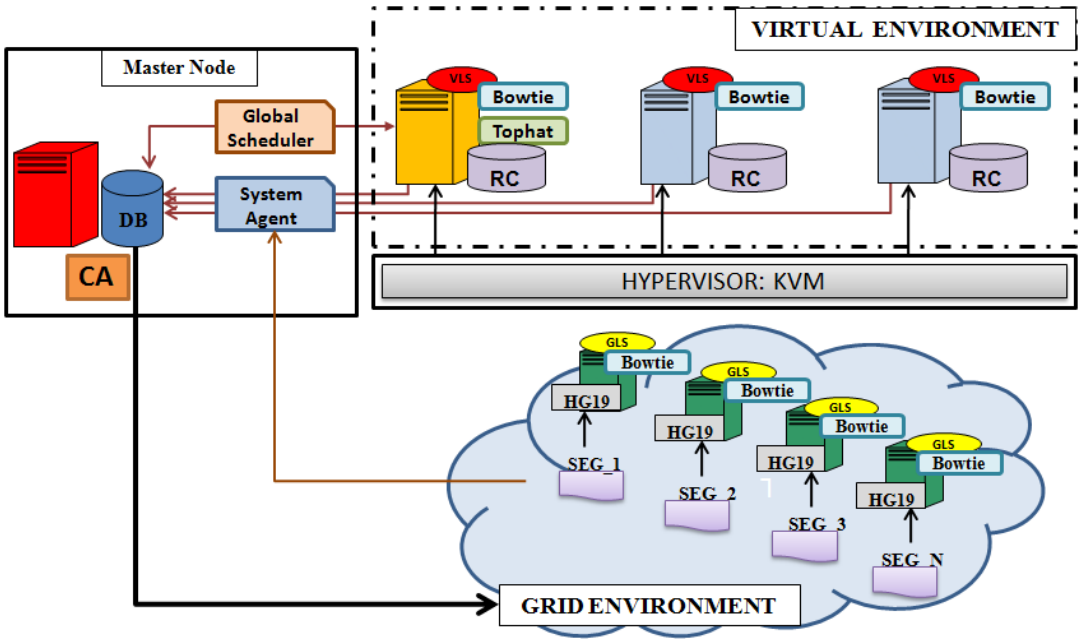

Figure 4. VirtualBio Architecture.

flexible component based structure and includes several pre-defined drivers for information management (monitoring), image and storage management (e.g., via NFS, LVM, or SSH), as well as to support several hypervisors (KVM, Xen, VMware). OpenNebula also integrates drivers for Amazon EC2 and ElasticHosts, and can be easily extended to support other cloud providers. With its support for EC2, OpenNebula can be configured to use Amazons infrastructure for cloud bursting, or use other EC2 compatible systems (such as Eucalyptus or Nimbus). An OpenNebula cloud typically consists of a front-end node for administration purposes (i.e., for managing hosts and images), as well as of several cluster nodes to execute the VM images. The hosts are controlled by the front-end either via command-line tools, or via well-defined programming interfaces.

1) Grid Environment: The Grid environment consists of machines with high computing power, this allows to use own machines and also machines belonging to different virtual organization. The only requirement is to have the necessary software installed for the processing (Bowtie, TopHat and Globus Toolkit). On each worker node of the grid environment is installed the Grid Local Scheduler, an essential component for performing biological tests.

2) Grid Local Scheduler: The Local Grid Scheduler (GLS) is a scheduler of active physical machines, which has been developed for the design phase (b), it aligns the segment with respect to the human genome (HG19) through Bowtie. Since the transfer of the input file is not influential, the worker nodes do not need to be in the same subnet as the Master Node, but may also belong to different virtual organization, so system can have greater scalability and can use machines powerful performance [10].

3) Virtual Environment: Virtualized environment also helps to improve infrastructure management, allowing the use of virtual node template to create virtual nodes in a short time, speeding up the integration of new nodes on the grid and, therefore, improving the reactivity and the scalability of the infrastructure [12]. The open source KVM has been used as hypervisor. It allows to create Fully Virtualized machines. The kernel component of KVM is included in mainline Linux [16]. KVM allows a Full Virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). KVM is implemented as a module within the Linux kernel. A hypervisor hosts the virtual machine images as regular Linux processes, so that each virtual machine image can use all of the features of the Linux kernel, including hardware, security, storage and applications. Full Virtualization provides emulation of the underlying platform on which a guest operating system and application set run without modifications and unaware that the platform is virtualized (see Figure 4). It implies that every platform device is emulated with enough details to permit the guest OS to manipulate them at their native level. Moreover, it allows administrators to create guests that use different operating systems. These guests have no knowledge about the host OS since they are not aware that the hardware they see is not real but emulated. The guests, however, require real computing resources from the host, so they use a hypervisor to coordinate instructions to the CPU. The main advantage of this paradigm concerns the ability to run

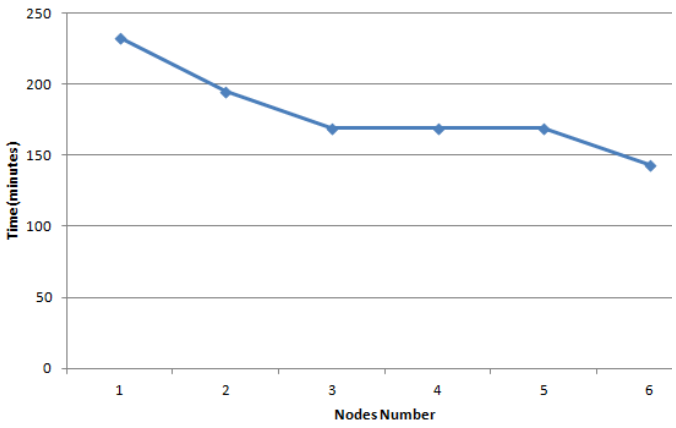

Figure 5. TopHat Execution Time for a Single Sample.

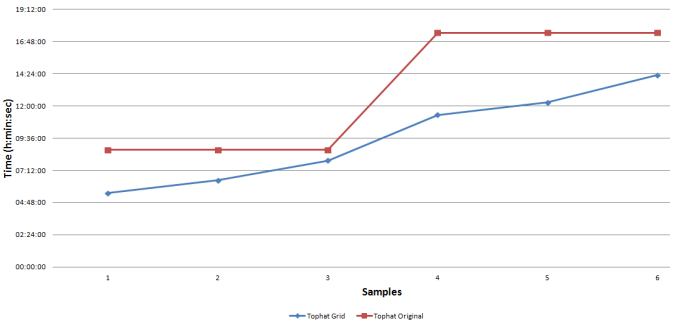

Figure 6. Original TopHat vs TopHat Grid.

virtual machines on all popular operating systems without requiring them to be modified since the emulated hardware is completely transparent. The virtualized environment has pre-installed images, which contain all software and libraries needed for running Bowtie and TopHat. The pre-configured images allow an ease instantiation of the machines when needed, and can be easily shutdown after the use.

4) Virtual Local Scheduler: The Virtual Local Scheduler (VLS) is a scheduler of active virtual machines. Its purpose is to draw up the steps (a) and (c) of TopHat. As the GLS, the VLS performs the mapping files for input received through Bowtie. The step (a) allows the alignment with respect to the human genome (HG19) and step (c) allows the alignment with respect to the fragment juncs previously constituted by TopHat. Since the considerable size of the files involved in these two steps, the VLS works directly on the temporary folder that is located in the common repository, allowing to avoid wasting time due to the transfer of data. Even in this case the interaction with the database is essential and very frequent, network problems may affect the entire biological analysis.

VI. Performance Considerations

In a preliminary work, we have introduced two case studies, based on Bowtie execution, from two different points of view. The first is the fragment size, while the other one is the CPUs number on the worker node. In Table I a summary of the calculations obtained changing CPUs number are presented. We can notice for reads between 100 and 1000000 no gain of time has occurred, so for our studies only reads from 1000000 to 85000000 are considered. This is because the files have limited data, thus the processing times are already reduced at this stage and then having multiple processors is irrelevant. As we explained before, during an analysis phase of the algorithm, 3 main blocks have been identified, (a) left and right mate aligned with HG19, (b) segments aligned with HG19, (c) segments aligned with segment juncs. The processing time of each segment depends on parameter pthread that is specified in command of Bowtie and refers to the number of parallel processes that can run. Figure 5 depicts the processing time of entire flow of TopHat for a single sample when nodes number increases. Elaboration with only a node corresponds to original version of TopHat, it means in sequential version. We want to focus the attention on elaboration time when 3/4/5 nodes are available, we obtained no gain of time because each node has more than one segment to process.

For this test phase, we wanted to use an architecture which consists of three machines with four CPUs. In Figure 6, we can notice that already only a sample processed with the version of TopHat Grid, a time savings of 40% is obtained instead increasing the number of samples to be processed, it is worth noting that the percentage of earned time is about 30%, this is due to the jobs queues that are created on the nodes. In Figure 7, processing time of a single segment of the variation of the parameter pthread is depicted. The processing time of each segment depends on parameter pthread that is specified in command of Bowtie and refers to the number of parallel processes that can run. Once past this threshold, the trend is no longer regular, this is due to the scheduling allocation of the CPU operating system. This test allowed to have a vision on the processing time will have access to machines with different power, opening to a more accurate scheduling policy adapted to the needs of time of the biologist. The test was run on a machine with 12 CPUs, it is worth noting that in order to gain the maximum time the number of pthread must be equal to the number of CPUs.

In Figure 7, processing time of a single segment of the variation of the parameter pthread is depicted. The test was run on a machine with the following hardware characteristics: Intel Xeon CPU X5660 @ 2.80 GHz, 12 CPUs and 20 GB of RAM, it is worth noting that in order to gain the maximum time the number of pthread must be equal to the number of CPUs. Once past this threshold, the trend is no longer regular, this is due to the scheduling allocation of the CPU operating system. This test allowed to have a vision on the processing time will have access to machines

Table I
BOWTIE EXECUTION TIME (SECONDS)

Reads	1 CPU	2 CPU	3 CPU	4 CPU	5 CPU	6 CPU	7 CPU	8 CPU
100	31	24	24	24	24	25	32	26
1000	26	38	24	24	24	25	13	20
10000	24	24	22	23	22	23	22	22
100000	28	25	25	24	24	24	24	24
1 E06	62	49	50	42	41	41	38	38
10 E06	424	258	230	185	176	165	158	154
85 E06	3425	2044	1791	1432	1342	1247	1180	1048

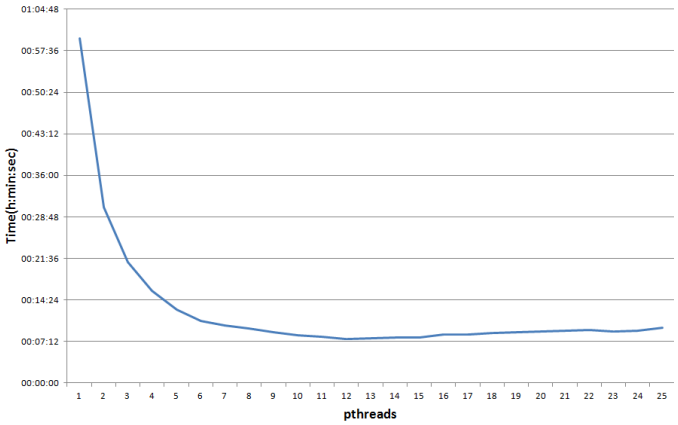

Figure 7. Bowtie Execution Time.

with different power and CPUs number, opening to a more accurate scheduling policy adapted to the needs of time of the biologist. Instead, Table I depicts the processing time of entire flow of TopHat, comparing the original algorithm (sequential version) with the modified algorithm (parallel version). We obtain a considerable gain of time that varies depending on the power of machines available. In our tests in order to make homogenous system, we have used machines with the same hardware (12 CPUs).

A. Amazon testbed

For the performed tests on Amazon EC2 was used a virtual machine High-Memory Quadruple Extra Large Instance (m2.4xlarge) [17] with the following characteristics:

- 68.4 GB of memory RAM;
- 26 EC2 Compute Units (8 virtual cores with 3.25 EC2 Compute Units each);
- 100 GB of instance storage;
- 64-bit platform;
- Cost $2,28/hour.

This virtual machine has been added to the worker nodes and has been personalized installing Bowtie and all grid services necessary to enclose this node in the Virtual Grid. The executed tests on this machine show that although there is a decrease of the execution time (was used an 8 virtual cores instance) the transfer time increases however. Table II shows the time necessary to execute 6 fragments and the total transfer time. The input files for each fragment are 500 MB and the output files are 1.2 GB.

The usage costs for this instance were $6.84. It is noted that the transfer time is very high, almost half of the total time. To resolve this problem there are two possibilities: move all the Virtual Grid Infrastructure on Amazon, but the usage costs would increase or split the fragments to be processed further.

Table II
AMAZON'S INSTANCE USAGE TIME

Total Execution Time	Total Transfer Time INPUT Files	Total Transfer Time OUTPUT Files	Total Usage Time
01:12:00	00:24:00	00:48:00	2:24:00

VII. CONCLUSION AND FUTURE WORKS

VirtualBio is a platform for NGS analysis, with particular focus to the read alignment process using TopHat and Bowtie. The authors want to offer to biologist a private hybrid infrastructure to conduct their studies. After a careful study of existing solutions and state of the art such as Myrna, Supersplat and SplitSeek, we designed and developed the proposed solution. The novelty of this solution covers two aspects: computational infrastructure and optimization algorithm of TopHat. The hybrid infrastructure is composed of two environments: one based on grid computing and the other one virtual environment and with implementation of a common repository and a set of job schedulers. The second novelty covers algorithm aspects, during reverse engineering code of TopHat algorithm, the workflow has been optimized making parallel tasks that before were sequential. The proposed architecture allowed to reduce the elaboration time by at least 40% without additional cost due to the purchase of hardware. Future works include the optimization of scheduling policies opening the way to a multisample scenario and implementation of the architecture in Cloud environment, thus increasing the system scalability. It can include the integration of this solution in European Grid Infrastructure in order to able to submit tasks across the Europe

REFERENCES

[1] O. Terzo, L. Mossucca, A. Acquaviva, F. Abate and R. Provenzano, *Virtual Environment for Next Generation Sequencing Analysis*, The Eighth International Conference on Networking and Services (ICNS 2012), St. Maarten, Netherlands Antilles, 25-30 March 2012, IARIA, pp. 47-51

[2] De Magalhes J.P., Finch C.E. and Janssens G., *Next-generation sequencing in aging research: Emerging applications, problems, pitfalls and possible solutions.*, Ageing Research Reviews, 2010 Jul; Vol. 9(3), pp. 315-323

[3] Sanger F., Nicklen S. and Coulson A.R., *DNA sequencing with chain-terminating inhibitors*, Proc. Natl. Acad. Sci. USA 74, 1977), pp. 5463-5467

[4] Pop M. and S.L. Salzberg, *Bioinformatics challenges of new sequencing technology*, Trends Genet. Vol. 24, 2008

[5] Kircher M. and Kelso J., *High-throughput DNA sequencing concepts and limitations.*, Bioessays. 2010 Jun, Vol. 32(6), pp. 524-356

[6] Trapnell C., Pachter L. and Salzberg Steven L. *TopHat: discovering splice junctions with RNA-Seq*, Bioinformatics (2009), Vol. 25, Available at: doi:10.1093/bioinformatics/btp120, pp. 1105-1111

[7] Langmead B., Trapnell C., Pop M. and Salzberg Steven L. *Ultrafast and memory-efficient alignment of short DNA sequences to the human genome*. Genome Biology 10:R25

[8] Maher C.A., Palanisamy N., Brenner J.C., Cao X., Kalyana-Sundaram S., Luo S, Khrebtukova I., Barrette T.R., Grasso C., Yu J., Lonigro R.J., Schroth G., Kumar-Sinha C., Chinnaiyan Y., *Chimeric transcript discovery by paired-end transcriptome sequencing*, AM. Proc Natl Acad Sci USA, 2009 July, Vol. 28

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

[10] Kurowski K., Nabrzyski J., Oleksiak A. and Weglarz J. *Scheduling jobs on the Grid-Multicriteria approach*, Computational Methods in Science and Technology, 12(2), pp. 123-138, 2006

[11] Amazon Elastic Compute Cloud (Amazon EC2) Available at: http://aws.amazon.com/ec2/, January, 2012

[12] A. Chierici and R. Veraldi, *A quantitative comparison between xen and kvm*, Proc. of 17th International Conference on Computing in High Energu and Nuclear Physics, Journal of Physics: Conference Series 219 (2010).

[13] Network File System, http://wiki.ubuntu-it.org/Server/Nfs, December, 2011

[14] Globus Toolkit, Available at: http://www.globus.org/toolkit/, January, 2012

[15] OpenNebula, Available at: http://opennebula.org/, January, 2012

[16] Kernel-based Virtual Machine, Available at: http://www.linux-kvm.org/, December, 2011

[17] AWS Amazon, Available at: http://aws.amazon.com/ec2/instance-types/, January, 2012