PhD Dissertation

International Doctorate School in Information and Communication Technologies

DIT - University of Trento

A high performance computational environment for UHTS studies.

Silvano Paoli

Advisor:

Dr. Cesare Furlanello

Fondazione Bruno Kessler

March 2010

Abstract

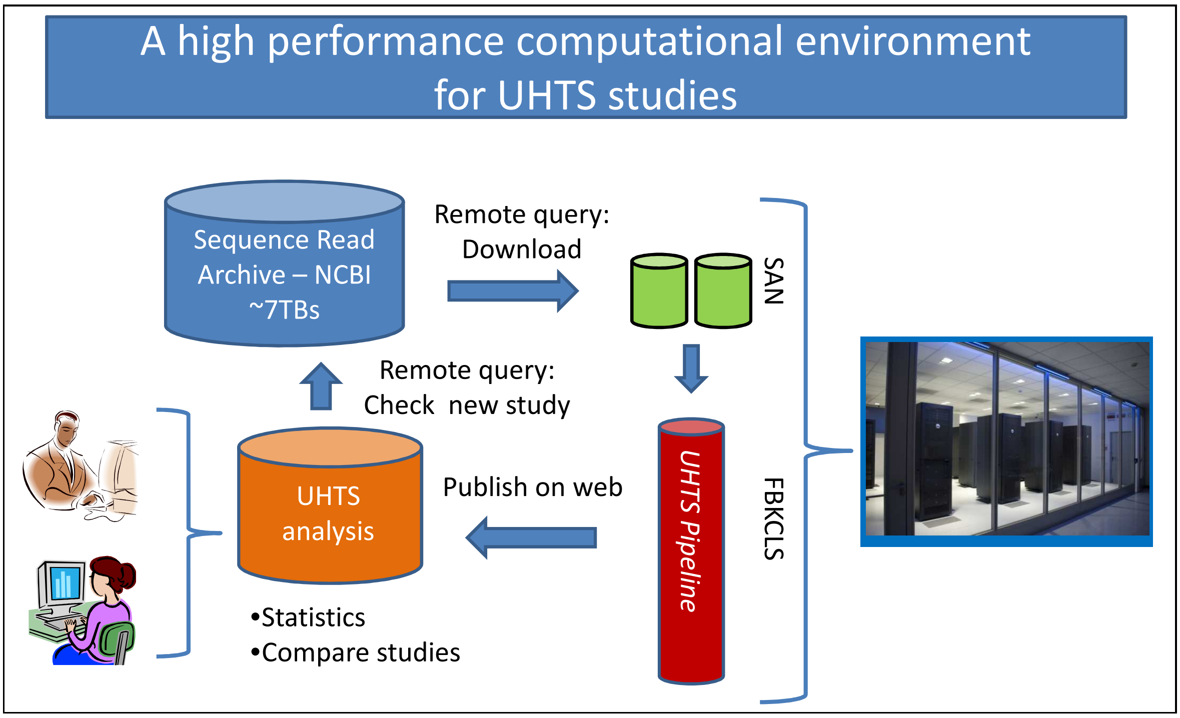
This work regards the use of *high performance computing* (HPC) methods for a new bioinformatics challenge: the analysis of Terabyte-size data generated by the new *ultra high throughput sequencing* (UHTS) technology.

As in microarray or mass spectrometry cases, *public repositories* are growing to store data from the next generation studies produced in laboratories around the world. These can be used to access to a large number of samples from experiments with different individuals, populations and sequencing platforms. Also experimental data of scientific articles are published in these stores, enabling to repeat and verify their results (reproducibility).

An *automatic downloader and analyzer system (D-Daemons architecture)* is proposed to interface to a public repository of sequence reads, select all the experiments that match some research parameters, defined by a user, download them and apply an analysis pipeline to evidence their similarity or variability. A software pipeline based on this architecture and operating in a *HPC environment* has been developed to analyze the downloaded UHTS files in the shortest time possible. A case study of the system on "Colorectal Cancer (CRC) cell line" datasets and an aligner selection in a SNP discovery task on three RNA-Seq datasets (Human Breast tissue and of BT474 & MCF7 cell lines) are presented.

Keywords

HPC, UHTS study, public repositories, software pipeline.



A high performance computational environment for UHTS studies

Sequence Read Archive – NCBI ~7TBs

Remote query: Download

SAN

Remote query: Check new study

UHTS analysis

Publish on web

UHTS Pipeline

FBKCLS

- Statistics
- Compare studies

The automatic downloader and analyzer system.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Context

High-Performance Computing (HPC) is the discipline of using supercomputers, computer clusters, and special purpose computational systems to solve problems that require high processing capacity, and in particular speed of calculation. HPC methods have been systematically employed for the analysis of huge volumes of data generated in bioinformatics tasks. In particular, the use of HPC has been the heart of the Human Genome Project, i.e. the sequencing of the human genome (1990-2000), whose goal was to determine the sequence of chemical base pairs defining the DNA. HPC facilities based on supercomputers allowed to complete the sequencing and assembly of the about 3 billion bases of the genetic code.

This thesis regards the use of HPC methods for a new bioinformatics challenge: the analysis of Terabyte-size data generated by the new *ultra high throughput sequencing (UHTS)* technology.

1.2 Next generation sequencing

DNA sequencing methods are the keystones of basic biological research and of most molecular medicine studies. They include all the sequencing methods for revealing the order of the nucleotide bases (adenine, guanine, cytosine, and thymine) in a molecule of DNA. The chemical sequencing (also known as Maxam-Gilbert sequencing) introduced the first generation of methods in 1977. Very soon the alternative technology of *chain terminator sequencing* or *Sanger sequencing* was developed by Frederick Sanger, Nobel Prize in 1980, allowing sequence length sequence length of 800-1000 nucleotide bases (or base pair long: *bp*) through several hours of chemical reactions. Since the Sanger method is relatively simple and cheap, so it is still widely used for small-scale experiments. In recent years, great efforts have been spent to improve the efficiency of DNA sequencing. *Next generation sequencing (NGS)* technologies have upgraded first with large scale methods and more recently with a new group of methods (*ultra high throughput sequencing - UHTS*) designed to parallelize the process and produce from thousands to billions of *short reads* at once, ranging from tens to one-two hundred base pairs.

Different sequencing platforms are available. Solexa Illumina, SOLiD and Roche 454 are the major of next generation sequencing technologies used in UHTS studies.

Vendor	Platform	Read Length
Illumina	Genome Analyzer IIx, IIe	35,50,75,100 bp
	HiSeq 2000	35,50,100 bp
Roche	454 GS FLX	200-300 bp
	454 GS FLX Titanium	400 bp
Applied Biosystem	SOLiD 3 System	50 bp
	SOLiD 4 System	50-100 bp
Helicos	HeliScope	25-55 bp

Table 1.1: Example of read lengths in UHTS platforms.

RNA-Seq analysis

The possibility to produce millions of sequences from a sample provides different methods to better understand the pathways involved in biological processes, e.g. the interactions between genes and proteins. Based on modern sequencing methods, the *UHTS transcriptome analysis* allows a low level survey of the produced gene activity in a cell. The transcriptome is the set of all *RNA* (mRNA, rRNA, tRNA, ncRNA) (see Section 2.1) molecules (also called "transcripts") produced in a cell. While the genome is roughly stable in cells, the transcriptome can vary during life due to a number of factors. Because it includes all transcripted mRNA in the cell, the transcriptome reflects the genes that are actively expressed at a one given time. Transcriptomics examines the expression level of mRNAs in a given cell population. *RNA-Seq* is an approach to study the transcriptome level on a biological sample using UHTS technologies.

In general a RNA-Seq experiment includes the following steps. First, one sample is sequenced by an UHTS machine. Then, the resulting short reads have to be compared with a *reference genome* sequence of the sample's species under investigation. In a reference genome, an organism is described through chains of nucleotide bases which form each chromosome. Each short read is compared to the chains in the reference genome. This search tries to find regions in chromosome sequences where a short read's nucleotide base sequence matches. The comparison is performed by looking at each nucleotide base in the reference genome with the succession of bases in the short reads.

Chromosomes contain genes which are involved in protein synthesis. The knowledge of all genes and their functions is still not complete for the human genome (and in other species). However a large set of genes is known and a map of them on a reference genome is available (see HG18

[2]). The positions of genes on chromosomes sequences allow us to check if a short read falls within these regions. If more than one short read falls on the region corresponding to one gene, we can suppose that this gene is expressed in the studied tissue, according to the association between the messenger RNAs (sequenced in short reads) and the gene. The number of short reads aligned over the coordinates of a gene gives a measure of the expression level of this gene. A peak of several short reads over a gene region shows that this gene is heavily expressed. Although the knowledge of the entire human genome is incomplete, the RNA-Seq analysis can then help to fill the gaps of knowledge, because pile of short reads in a position of a chromosome where no known gene occurs can encourage to investigate existence of a new element, while a short read stack over the boundaries of a gene's region addresses to rethink to the real distribution of this gene in the chromosome at least in terms of individual variability. Furthermore, this detailed knowledge can be used for functional genomics, replicating in the UHTS framework all the signal evaluation until now provided in high throughput by array technologies. This thesis aims to provide new computational tools to automate the RNA-seq analysis in a wide range of problems such as more briefly introduced in this section.

Software tools for UHTS studies

In parallel with the development of the UHTS technologies, several software tools have been developed to organize data and control the alignment of short reads to a reference genome. The first software was written and sold by vendors of the sequencing machines. Very soon, the bioinformatics community started to develop Open Source Software to replace and improve vendors' instruments. From a computer science point of view, the final major task with RNA-Seq data is to align reads to a reference genome and to perform statistics of where, how and how many reads fall on genes and chromosomes in genome. In a RNA-Seq experiment applied on human tissue, million of reads can be sequenced and have to be aligned to a human reference genome. Each read and the reference genome are considered as strings of characters in the computer's memory; each string is composed by long combinations of four letters (A,C,G,T), which represent the four nucleotide bases (A=adenine, C=cytosine, G=guanine and T=thymine). While short reads have length of tens or hundreds of characters, a reference genome is described by millions or billions of characters. For example, the human reference genome HG18 ([2]) has more than 3 billions of bases. The alignment phase of short reads on a reference genome is translated in a computer science task called *string matching problem*. The first step is to find a place where one or several strings (short reads) are found within a larger string (reference genome). Several *string searching algorithms* [3] were developed in computer science. The first generation of sequence alignment tools were based on hashing, but the enormous memory requirements for the hash table was a major drawback. The following generation of alignment programs uses the *Burrows Wheeler Transform* (BWT) [3] to efficiently align short sequencing reads against a large reference sequence such as the human genome. Programs of the first generation include Eland (a module within the Illumina software suit, provided as free source code for Illumina's machine buyers), SOAP [4] (freeware) or MAQ [5] (Open Source code). Examples of new generation Open Source programs are Bowtie [3] and BWA [6], while SOAP2 [7] is freeware.

After the alignment phase, an investigation to count how many reads fall on the different chromosomes and genes is imperative to understand the transcriptome activity of the sequenced tissue sample. We can indicate this process as post-processing alignment phase. Typically, genes are divided in zones called exons and introns. An exon is a DNA chain in a gene which takes part in the composition of the coding sequence used to build a protein. An intron is a DNA section within a gene that is not translated into a protein. Each gene has a variable number of introns and exons (inside its interval on chromosome). Knowing how many reads fall into exon zones allows one to understand the *RNA splicing* in a gene. This is a mechanism in the transcription phase to create the final RNA code from the gene's DNA sequence, which will be used to produce a protein. Here, the gene's exons are recombined and joined together to produce the RNA code. Another key goal is to find statistically significant peaks of short reads to examine interesting zones on chromosomes. The expression level of a gene can be weighed through the RPKM measure, i.e. reads per (kilo)base per number of mapped reads. A single nucleotide polymorphism (SNP) detection can be performed on mapped short reads. A SNP is a single base pair mutation at a specific location of a gene or a DNA sequence on a chromosome. It allows a measure of the genetic variation between samples, specifically in a set of disease and normal samples from the same tissue. Open Source programs as TopHat [8] and Erange [9] can perform some of the previous analysis, given a set of aligned short reads. A complete RNA-Seq experiment requires both an alignment and a post-processing alignment phase. In almost all cases, more than one software has to be applied to obtain a complete result.

Public repositories of UHTS data

The number of UHTS studies being conducted around the world by teams of biologists and bioinformaticians is quickly increasing. Great scientific advance is due to the increasing sharing of UHTS data. As for microarray and proteomics data, public repositories for Ultra High Throughput Sequencing data are growing in the web. They allow sharing data used in UHTS studies to replicate results, to apply new techniques or to benchmark software tools. One example of these new repositories is the *Sequence Read Archive* (SRA) [10]. The National Center for Biotechnology Information (NCBI) maintains the SRA as a repository for data from sequencing projects that use the new massively parallel sequencing technologies. The Sequence Read Archive will become more and more important as next generation sequencing technologies continue to improve and become even less expensive. The SRA accepts and presents data from all current next generation sequencing platforms including 454, Illumina, SOLiD, HeliScope, and Complete Genomics. Individual read lengths vary from around 25 bases to more than 400 bases depending on the platform. Data can include sequence, quality scores, color values, and intensity graphs depending on the platform involved.

1.3 A novel automated pipeline

Here goes a short introduction to the technical issues motivating the solution proposed in this thesis. The UHTS studies allow a better understanding of the processes in transcriptome phases. In parallel to this technology an interesting set of software has grown to analyze and archive raw sequences produced from the new parallel sequencing machines. A RNA-Seq experiment can produce millions of short reads, which have to be aligned on a reference genome. Two computer science problems can be found in this task. The first is to archive and manage the short reads, because these can require tens of gigabytes for just a single sample. The second is to perform alignment and post-processing alignment procedures in the most efficient way to increment the number of analyzed short reads and to reduce the time to have a complete result. The main hurdle remains the computational burden of a cycle of analysis, currently higher than one week of computation with a large set of short reads on human data on a standard workstation.

The proposed solution is to select the state-of-art software in UHTS studies, order them in a workflow and run it in a *High Performance Computing* environment.

High Performance Computing is normally obtained by deploying environments endowed with large numbers of CPUs, in which a serial software can be split in several instances simultaneously run to reduce the computational time. Heavy computational tasks can be efficiently managed on HPC systems, initially implemented through large machines with exotic architectures. Modern HPC facilities are based on the concept of *cluster computing*, that is a group of linked workstations (also called nodes) which work together. The current multi-core architectures in CPU technology further helps to add computing power within limited space, thus allowing to a huge number of elaboration cores to be setup in the computing environment. In HPC facilities, a further key element is the *queue system* that coordinates users and allows the distribution of users' programs on available computational nodes.

The UHTS software, especially for alignment of short reads, are developed to run on a single machine. Few of them can run on a multi-core workstation, splitting their algorithms on several threads. While, this type of parallelization guarantees a little speed-up, it can work on only one machine. To enable an efficient access to HPC environments, this software has to run as a chain of software elements (*pipeline*) suited with the queue system of environment. This pipeline can link together different tools and distribute the workload on several nodes for heavy computational phases (as alignment of short reads).

1.4 Innovative Aspects

This thesis aims to combine state-of-art Open Source Software used in RNA-Seq studies and to develop a new HPC-based platform that may greatly accelerate the overall performance. Differences and causes of variability in the process have been considered to select the best elements for a *pipeline of software tools* and to guarantee the following features:

1. This pipeline should reach the maximum productivity in the analysis of up to 100 samples, each one consisting of millions of short reads.

2. The pipeline should be designed to operate on *High Performance Facilities* (HPC), as cluster computers, to parallelize and speed-up different steps. A particular attention has been dedicated at evaluating and modifying software tools compatible with a HPC environment.

3. Standard measures, statistics and formats applied to raw sequences and alignment results are considered.

4. Finally, the possibility to insert the proposed pipeline in an automatic download system from the SRA repository is examined under the architecture point-of-view.

1.5 Structure of the Thesis

The rest of this thesis is composed the following chapters:

Chapter 2 presents some basic definitions in UHTS and RNA-Seq contexts, describes analysis steps in a standard study and reports the state-of-art Open Source Software used in an ultra high throughput sequencing investigation;

Chapter 3 analyzes the technical goal of this work and explains the D-Daemons solution;

Chapter 4 gives some experimental results on components of an analysis pipeline;

Chapter 5 presents the issue of reproducibility, providing two examples of analysis through proteomics pipelines;

Chapter 6 provide a review of related work;

Chapter 7 summarizes the problem and solution and discusses possible developments of the presented work.

Thesis is completed by a Reference section and Appendix listing the academic and technical papers produced in the context of the PhD study.

Chapter 2

State of the Art

2.1 Transcriptomics

In life forms the operations in a cell are driven by the *proteins.* These are essential parts of organisms and participate in virtually every process within cells. How the proteins work is the key to understand the evolution steps in a cell during its life cycle. This comprehension includes the cell's behaviors when a disease attacks an organism, in particular this information can help to understand serious diseases like cancers and tumors. The proteins are complex objects and the study of their interactions is not a simple task. The *central dogma of the molecular biology* shows an alternative way to understand these processes in a cell. It states that in synthesizing proteins, DNA is transcribed into mRNA, which is translated into proteins. So, in place of investigating the proteins directly, it is possible to study the mRNA and the genes, where these chains of nucleotide are transcribed, involved in the synthesis process.

The transcriptome is the set of all RNA molecules, including mRNA, rRNA, tRNA [1], and non-coding RNA produced in one or a population of cells. The term can be applied to the total set of transcripts in a given organism, or to the specific subset of transcripts present in a particular

[1] Ribonucleic acid (RNA), Messenger ribonucleic acid (mRNA), Transfer ribonucleic acid (tRNA).

cell type. Unlike the genome, which is roughly fixed for a given cell line (excluding mutations), the transcriptome can vary with external environmental conditions. Since it includes all mRNA transcripts in the cell, the transcriptome reflects the genes that are being actively expressed at any given time. *Expression profiling* is referred to studies of transcriptomics where the expression level of mRNAs in a given cell population is examined; expression measures are generally provided through high throughput techniques based on DNA microarray technology. The use of the new Ultra High Throughput Sequencing technology allows determining the order of the nucleotide bases (adenine, guanine, cytosine, and thymine) in RNA molecules. The study of the transcriptome at the nucleotide level is known as *RNA-Seq*.

Thanks to the deep coverage and base level resolution provided by next generation sequencing instruments, RNA-Seq provides researchers with efficient ways to measure transcriptome data experimentally, allowing them to get information such as how different alleles or alternative forms of a gene are expressed, detecting post-transcriptional mutations or identifying gene fusions. RNA-Seq is a technique that is quickly becoming new reference in the study of diseases like cancer.

In a typical RNA-Seq experiment, the sequencing process generates millions of small chains of nucleotide bases. These sequences are long from tens of bases to one or two hundreds bases. The length depends by the specific sequencing platform (vendor). For example the Illumina platform uses 35, 50, 75 and 100 bases and the Roche 454 ranges from 200 to 400 bases (see 1.1). These sequences are called *short reads*. The nucleotide series have to be aligned on a *reference genome*, which is the set of the known nucleotide sequences that compose the chromosomes. The position of each gene is defined on the nucleotide chain of one chromosome, so it is possible to count how many short reads fall over or close to one gene. By counting how many short reads from the RNA molecules of a sample coverage determinate positions on the chromosomes, we can gather information on the expressed genes for this sample. We will use this oversimplified view of the whole process to describe the bioinformatics tools of interest in this context.

2.2 Advanced software tools

All ultra high throughput platforms save short read sequences as text strings in large files. The letters used in these strings are usually “A,C,G,T”, which represent the four nucleotide bases (A=adenine, C=cytosine, G=guanine and T=thymine) or an alternative encoding defining the so-called “color space”. Given this computing science point of view of the sequencing process, several software tool have been developed to manage sequencing experiments and to perform analyses, like alignment to the reference genome (which is also encoded as a huge text file). After the previous brief introduction on the transcriptomics, what follow is a description of the state of the art software tools used in UHTS (and RNA-Seq) studies.

2.2.1 Erange

The Enhanced Read Analysis of Gene Expression (ERANGE) [9] is a set of command-line Python scripts used to perform a RNA-Seq study. Each Erange script performs a function and they can be called sequentially to produce a complete analysis. The scripts save their analyses in a SQLite flat file database. Erange is not an aligner and uses external software to perform the alignment phase of short reads on reference genomes. The supported aligners are Eland, included in the commercial software package of the Illumina platform, and Bowtie, an open source aligner which will be described in the next sections. The Erange software package makes it possible to:

- Assign reads that uniquely map on the genome to their site of origin.
- Select reads that match equally well to several sites (called *multireads*) to their most likely site(s) of origin on the reference genome.
- Detect splice-crossing reads and assign them to their gene of origin.
- Organize reads that cluster together, but do not map to an already known exon, into candidate exons or parts of exons.
- Calculate the prevalence of transcripts from each known or newly proposed RNA, based on normalized counts of unique reads, spliced reads and multireads.

Erange is the first software introducing a new normalized measure of transcripted short reads on the reference genome: defined by the sensitivity of a RNA-Seq sample/experiment as a function of both molar concentration and transcript length. The Erange's writers quantify transcript levels in reads per kilobase of exon model per million mapped reads (RPKM). The RPKM [9] measure of read density defines a sort of "molar" concentration of a transcript in the starting sample by normalizing for RNA length and for the total read number in the measurement. This facilitates comparison of transcript levels both within and between samples and experiments.

2.2.2 Bowtie

Bowtie [3] is an open source alignment program for aligning short DNA sequence reads to large genomes. Bowtie uses a different and novel indexing strategy to create an ultrafast, memory-efficient short read aligner geared toward mammalian re-sequencing. Bowtie can sort out 35-base pair (bp) reads at a rate of more than 25 million reads per CPU-hour, which is more than 35 times faster than previous aligners. This software is part of the next generation of alignment programs because it is not based on hash-table methods and it avoids their enormous memory requirements. Indeed Bowtie employs a Burrows-Wheeler index, which guarantees a memory footprint of only about 1.3 gigabytes (GB) for the human genome. This smaller footprint allows Bowtie to run on a typical desktop computer with 2 GB of RAM. Moreover Bowtie can run on multiple processor cores simultaneously to achieve higher alignment speed.

The Burrows-Wheeler transform (BTW) is a reversible permutation of the characters in a text. Although originally developed within the context of data compression [3], BWT based indexing allows large texts to be searched efficiently in a small memory footprint. It has been applied to bioinformatics applications, including oligomer counting, whole-genome alignment, tiling microarray probe design, and Smith-Waterman alignment onto a human-sized reference [3].

In Bowtie, the BWT based index is built upon on the Ferragina and Manzini [3] exact-matching algorithm. Here, the procedure is not simply adopted as-is, because exact matching does not allow for sequencing errors or genetic variations. The program introduces two novel extensions that make the technique applicable to short read alignment: a quality-aware backtracking algorithm that allows for mismatches and favors high-quality alignments; and "double indexing", a strategy to avoid excessive backtracking. The Bowtie's policy allows for a small number of mismatches within the high-quality end of each read, and it places an upper limit on the sum of the quality values at mismatched alignment positions [3].

Bowtie can save alignment in the new standard SAM (Sequence Alignment/Map) format [11], which is used by other RNA-Seq software tools (see [6] and [11]).

2.2.3 BWA

The Burrows-Wheeler Alignment tool (BWA) [6] is the next generation version of the software called MAQ [5], based on the hash table-based methods, which is accurate, feature rich and fast enough to align short reads from a single individual. BWA is a new read alignment package that is based on backward search within BWT to efficiently align short sequencing reads against a large reference sequence such as the human genome, allowing mismatches and gaps. BWA supports both base space reads, e.g. from Illumina sequencing machines, and color space reads from AB SOLiD machines. Evaluations on both simulated and real data suggest that BWA is approximately 10-20x faster than MAQ, while achieving similar accuracy. In addition, BWA outputs alignment in the new standard SAM (Sequence Alignment/Map) format. Variant calling and other downstream analyses after the alignment can be achieved with the open source SAMtools software package [11].

2.2.4 Tophat

TopHat [8] is a free Open Source Software package which allows to discover *splice junction* in RNA-Seq samples.

Splice junctions are locations on a DNA sequence at which "superfluous" DNA is removed during the process of protein synthesis in higher organisms [12]. This process is named *splicing*. The problem can be posed to recognize, given a sequence of DNA, the boundaries between exons (the parts of the DNA sequence that are retained after splicing) and introns (the parts of the DNA sequence that are spliced out), which are called "donor" and "acceptor sites". This problem can also be redefined as to find a classification rule: given a position in the middle of a window of DNA sequence elements (nucleotides), decide whether this is an intron →

exon boundary, exon → intron, or neither.

TopHat can identify splice sites "ab initio" by large-scale mapping of RNA-Seq reads. It maps reads to splice sites in a mammalian genome at a rate of ~ 2.2 million reads per CPU hour, which is sufficient to process an entire RNA-Seq experiment in less than a day on a standard desktop computer. Rather than filtering out possible splice sites with a scoring scheme, TopHat aligns all sites, relying on an efficient 2-bit-per-base encoding and a data layout that effectively uses the cache on modern processors. This strategy works well in practice because TopHat first maps non-junction reads (those contained within exons) using Bowtie (described in 2.2.2). TopHat, as Bowtie, can run in multithread fashion on multi-core CPU architecture.

In discovering splice junctions, TopHat maps reads to the reference genome in two phases. In the first one, 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 (two, by default) in the $5'$-most s bases of the read. [2]

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 (70 by default). This policy is based on the empirical observation that the $5'$ end of a read contains fewer sequencing errors than the $3'$ end. (Hillier et al., 2008). TopHat allows Bowtie to report more than one alignment for a read (default = 10), and suppresses all alignments above 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. Low complexity

[2] The $5'$ and $3'$ (usually pronounced "five prime end" and "three prime end") indicate the *directionality* in molecular biology and biochemistry and refer to the end-to-end chemical orientation of a single strand of nucleic acid. Single strands of DNA and RNA sequences are written in $5'$ to $3'$ direction.

reads are not included in the set of the “initially unmapped reads” reads: they are simply discarded. TopHat then assembles the mapped reads using the assembly module in MAQ [5]. TopHat extracts the sequences for the resulting islands of contiguous sequence from the sparse consensus, inferring them to be putative exons. To generate the island sequences, Tophat uses MAQ which produces a compact consensus file containing called bases and the corresponding reference bases. The reference genome is used to call the base. 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 (default = 45 bp). To map reads to splice junctions, TopHat first enumerates all canonical donor and acceptor sites within the island sequences (as well as their reverse complements); next, it considers all pairings of these sites that could form canonical (GTAG) introns between neighboring (but not necessarily adjacent) islands. Each possible intron is checked against the “initially unmapped reads” for reads that span the splice junctions, as described below. By default, only potential introns longer than 70 bp and shorter than 20000 bp are examined.

2.2.5 SAMtools

The SAM (Sequence Alignment/Map) is a generic format for storing large nucleotide sequence alignments. The goals archived by this format are:

a. to save all the alignment information generated by various alignment programs;

b. to be easily created by alignment programs or converted from existing alignment formats;

c. to generate files of compact dimension;

d. to allow most operations on the alignment without loading the whole alignment into memory;

e. to index file by genomic position to efficiently retrieve all reads aligning to a locus.

SAMtools [11] is a library and software package for parsing and manipulating alignments in the SAM format. It also operates with BAM format, which is the binary version of the SAM [13]. This library can convert from other alignment formats, sort and merge alignments, call SNPs and show alignments in a text-based viewer. In an alignment of the 112Gbp Illumina GA data, SAMtools took about 10hours to convert from the MAQ format and 40 minutes to index with less than 30MB memory.

2.2.6 Cufflinks

Cufflinks [14] is a program that assembles aligned RNA-Seq reads into transcripts, estimates their abundances, and tests for differential expression and regulation transcriptome-wide. In RNA-Seq experiments, cDNA fragments are sequenced and mapped back to genes and ideally, individual transcripts. Properly normalized, the RNA-Seq fragment counts can be used as a measure of relative abundance of transcripts, and Cufflinks measures transcript abundances in Fragments Per Kilobase of exon per Million fragments mapped (FPKM), which is analogous to single-read “RPKM”.

The *alternative splicing* process involves RNA exons produced by transcription of a gene, which can be reattached in various ways during the RNA splicing phase. Different mRNAs are generated and all these can be translated into different *protein isoforms*[3]. The result of this activity is that a single gene can code for multiple proteins.

[3] Different forms of a protein produced from related genes or from alternative splicing of the same gene.

Cufflinks can estimate a transcript abundances. This task is performed assigning fragments to individual transcripts, but it is not easy because a read may align to numerous isoforms of the same gene. A statistical model [15] is applied on sequencing experiments. This model generate a likelihood to estimate the abundances of a set of transcripts given a set of fragments.

Cufflinks takes as input a file of alignments in SAM format, and reports transfrags in GTF (Gene transfer format [16]) format. Bowtie and TopHat support the SAM design, so their output can be passed to this tool.

2.2.7 Crossbow

Crossbow [17] is Open Source Software for cloud-computing that combines the aligner Bowtie and the *single-nucleotide polymorphism* (SNP) caller SOAPsnp [18]. Crossbow is distributed with a set of scripts, which allow to run this tool either on a local cluster or on a cluster rented through Amazon's Elastic Compute Cloud (EC2) [19] utility computing service [4].

SOAPsnp performs a SNP analysis. A single-nucleotide polymorphism is a DNA sequence variation occurring when a single nucleotide, as A, T, C, or G, in the genome (or other sequence), differs between members of a species (or between paired chromosomes in an individual). These variations in DNA sequences can manifest how individuals develop diseases and respond to external agents.

By taking advantage of commodity processors available via cloud computing services, Crossbow condenses over 1,000 hours of computation into a few hours without requiring the user to own or operate a computer cluster [17].

[4] *Cloud Computing* is a term in computer science used to indicate a modern view of the client-server model, where elements are connected through the Internet network. This model involves the provision of dynamically scalable and often virtualized resources as a service over the World Wide Web.

2.2.8 Genome Browser

The UCSC Genome Browser website [20, 2] contains the reference sequence and working draft assemblies for a large collection of reference genomes for different species. It also provides a portal to the Encyclopedia of DNA Elements (ENCODE) project [21], which aims to provide a more biologically informative representation of the human genome by using high-throughput methods to identify and catalogue the functional elements encoded.

The Genome Browser section allows users to zoom and scroll over chromosomes, showing the work of annotators worldwide. Moreover, the users can upload and display their custom tracks on the available reference genome. Users' data have to be written as tab-separated files using one of the formats supported like GFF, GTF, PSL, BED, bigBed, WIG, bigWig, BAM, MAF, and microarray (BED15). Erange and TopHat can export their results in Genome Browser compatible file formats.

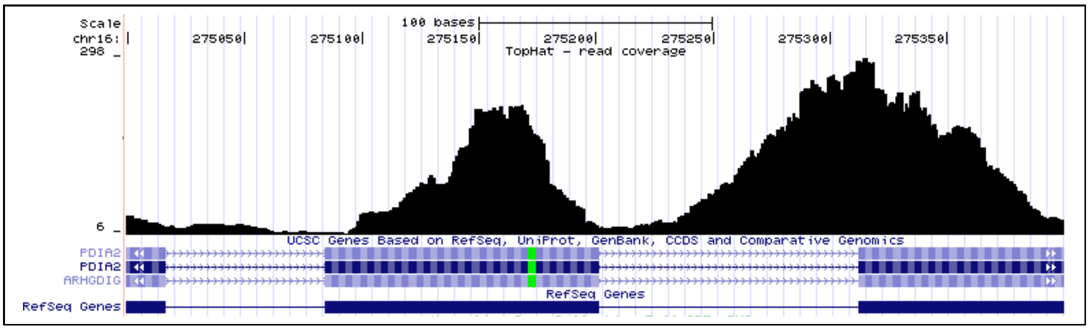


Figure 2.1: Example of alignments in Genome Browser from RNA-Seq Cerebellum (human) Illumina 35bp sample [1] over the HG18 reference genome

2.2.9 Galaxy

Galaxy [22] allows experimental biologists with no programming experience to locate and visualize genomic regions using intuitive graphical interfaces through a simple web browser. The system supports the integration of genomic sequences, their alignments, and functional annotation. Moreover, it allows users to gather and manipulate data from existing resources in a variety of ways. Every action of the user is recorded and stored in a history system. This mechanism allows users to run independent queries on genomic data from different sources and then use Galaxy to combine or refine them, perform calculations, or extract and visualize corresponding sequences or alignments.

The framework is written in Python language [23] and the data storage uses SQLite [24] a self-contained SQL database engine. Galaxy can also be configured to run jobs on a HPC cluster. One of the key features of Galaxy is its ability to obtain data directly from UCSC Table Browser. Any dataset in Galaxy's history can be displayed within UCSC Genome Browser as long as its chromosome, start, and end columns (so called metadata) are properly set and the corresponding genome assembly exists at the UCSC Browser.

2.2.10 SRA

The National Center for Biotechnology Information (NCBI), the European Bioinformatics Institute (EBI), and the DNA Data Bank of Japan (DDBJ) are involved in the International Nucleotide Sequencing Database Collaboration [25]. This collaboration has set up the Sequence Read Archive (SRA) [26, 27] to provide the scientific community with an archive for next generation data sets.

The resulting Sequence Read Archive (SRA) is now accessible from the NCBI [10, 26] , the European Bioinformatics Institute (EBI) [28] and from the DNA Data Bank of Japan (DDBJ) [29]. The three SRAs will mirror data and share an accession space, essentially providing a world-wide archive.

In November 2009, the SRAs collectively hosted about 11 Terabases of biological sequence data. This included 170 full-length human genomes, over 900 bacterial genomes, and 100 expression and epigenomics studies. Over 90 published studies have been linked to SRA deposits. The archive comprises platforms as 454, Illumina, SOLiD, Helicos, and others.

Most of the human genomes were produced by the 1000 Genomes Project [30], which is using sequencing data to perform a deep analysis of ordinary human variation in three healthy populations with the expectation of detecting common human genetic variants (defined as frequency 1% or higher). The goal of the 1000 Genomes Project is submitting reads to the SRA in real time as they are produced, allowing investigators not associated with this project to direct access to its output.

The value of the SRAs to the scientific community will depend on the degree to which data from investigations are deposited. Accordingly, NCBI, EBI and DDBJ encourage researchers to consider depositing their data in one of the SRAs.

Chapter 3

An automatic downloader and analyzer system

3.1 Problem

The chance to use the high accuracy information from the next generation sequencing technologies for a large number of samples is revolutionary step for molecular biomedicine. It can open to the possibility to investigate transcription events and gene dynamics through a numerical point of view. On very large set of experiments statistic measures and mathematical models can be applied to describe biology events. Tasks of interest could include the identification of signatures of disease in different individuals or the validation of models that can describe widespread features between different populations. The relatively small number of studies can be a constraint to this type of investigation. Fortunately, as in microarray or mass spectrometry cases, public repositories are growing to store data from the next generation studies produced in laboratories around the world. A repository can be used to access to a large number of samples from experiments with different individuals, populations and sequencing platforms.

The existence of these repositories is an invaluable resource for biologists and bioinformaticians. But how much are these sites usable in practice? A user has to connect to a repository, find his samples, download and process them. The last two phases are relatively complex operations in the UHTS case. First, as already described, the text files from a UHTS experiment can weigh several gigabytes, so a considerable time is required to move data from the repository to a local workstation. Second, the alignment phase and other post processing analyses can take several days and weeks when the number of short read files gets larger and the computational resource is only a single workstation. A manual download and exploration can be completed for a small amount of data files. Nevertheless, the possibility to retrieve from a large deposit of samples is a crucial step to involve investigations and comparisons between them.

The problem becomes how to manage the possible (few or many) heavyweight files (several gigabytes) from a public repository and to use them in a comparison study in a reasonable time.

This thesis tries to find a solution to this task. An automatic downloader and analyzer system is proposed to interface to a public repository of sequence reads, select all the experiments that match some research parameters, defined by a user, download them and apply an analysis pipeline to evidence their similarity or variability. The system is indicated as "automatic", because it can be scheduled after a period of time to find new files in the repository, which correspond to the selection. To reduce the waiting period, when the number of experiments is large, the solution includes the possibility to run the analysis pipeline on a high performance computing environment. The elements in the pipeline have been chosen from the state-of-the-art Open Source Software tools available in the UTHS described in Chapter 2 field and suited to operate in a HPC system. The public repository of next generation sequencing experiments chosen to operate with the developed architecture is the Sequence Read Archive (SRA) also described in Chapter 2. This archive has been selected because it stores a large number of sequencing trials and it is maintained by three international bioinformatics institutes which guarantee high quality in the recorded sequencing experiments.

3.2 Methods

The main goal is to build an automatic system that can process large numerical sets of ultra high throughput experiments, generating results that can be utilized to compare genomic attributes between samples.

The D-Daemon system is an automatic downloader and analyzer system that interfaces to the NCBI SRA. This public repository actually includes a huge number (11 Terabases) of high quality sequencing experiments. The D-Daemon system allows to select a sample in NCBI SRA on the basis of some user-defined parameters, downloaded and passed to an analysis pipeline.

Once activated, the system can be scheduled to find new data in the repository, corresponding to the user's parameters. To reduce the waiting time between harvesting of data and availability for biological analysis the solution is designed to operate an analysis pipeline in a HPC environment. The pipeline is composed by elements chosen from the state of the art and Open Source Software tools in the UTHS research community. For each experiment, the RPKM measure is calculated over all chromosomes. The sequences aligned on the reference genome are translated in a Genome Browser compliant file format for visualization. Finally the identification of new gene isoforms is supported.

All the procedures are written in the Python language. This choice is motivated by several reasons. Python is often used in bioinformatics field to build analysis tools [8] and to develop software structure or used as a glue to links together two or more systems [31]. It allows a rapid and efficient deployment of software infrastructure. Examples of bioinformatics platforms in this language can be found in [32] and [33].

As illustrated in Figure 3.1, the D-Daemon solution is composed by some steps that constitute the core procedures of the whole system. First of all (1) a remote query is performed to the SRA repository to discover the UTHS samples. (2) The files of each experiment are downloaded and passed to the HPC pipeline (3). Three tools have been wrapped to enable the possibility to run on the individual nodes of computer cluster infrastructure: Bowtie, Samtool and Cufflinks. From this analysis chain several measures are generated to evaluate and compare the samples. In the final step (4), outputs are formatted in HTML files to be published in a website. Some metadata information on downloaded and processed experiments is saved in a SQLite database. These metadata are used to look for new experiments in the SRA archive. If new data are available, they are downloaded, processed and published as described previously.

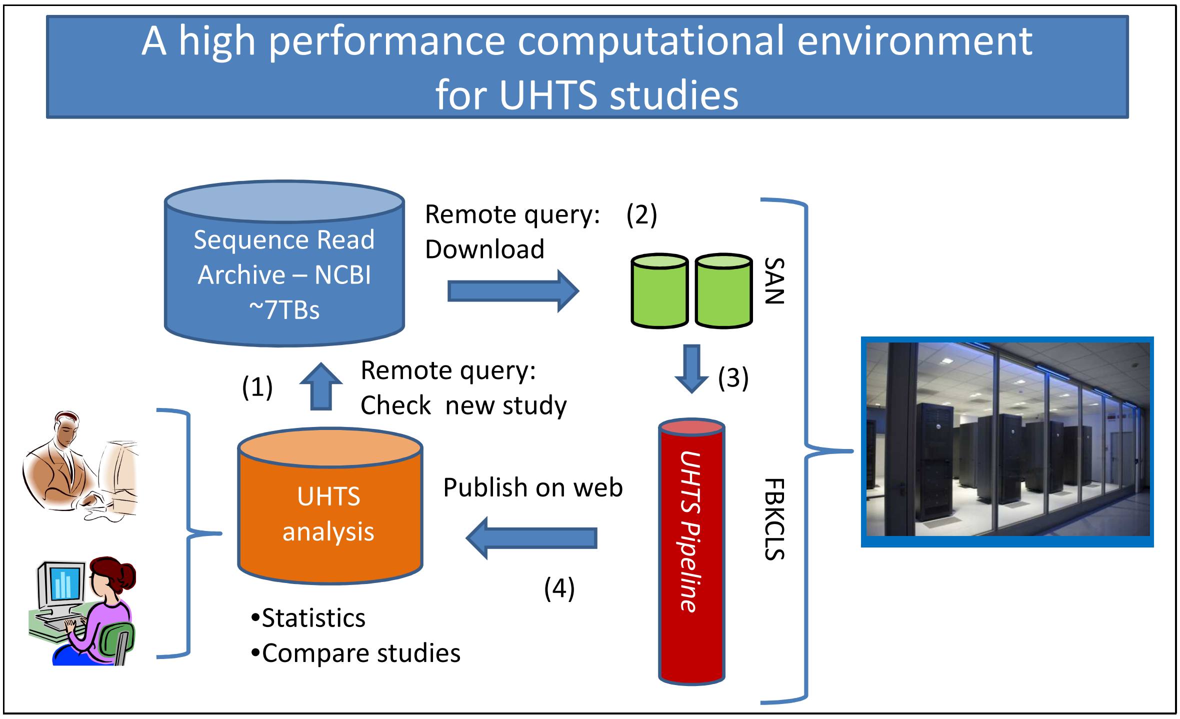


Figure 3.1: System description.

3.3 The D-Daemons architecture

Two modules or services constitute the infrastructure (see Figure 3.2) called *D-Daemons* architecture. The first is called *web daemon (web-dae)* and it checks the SRA repository to select the experiments, downloads them and registers their information in a SQLite database [24]. The web-dae also sends the experiment toward the HPC facility, where another service manages the HPC pipelines to process data. The web-dae withdraws the results and moves them to a web server, where details, RPKM values and Genome Browser files on the HG18 reference chromosomes are published for each experiment. This service is deputed to re-check the repository to new cases. The sample's characteristics, as tissue, platform, number of bases and others, are extracted from the *E*ntrez cross-database search, which is an integrated, text-based search and retrieval system used at NCBI for the major databases, including PubMed, Nucleotide and Protein Sequences, Protein Structures, Complete Genomes, Taxonomy and the Sequence read archive. The *Entrez Programming Utilities* [34] are tools that provide access to Entrez data outside of the regular web query interface and may be helpful for retrieving search results for future use in another environment. One of these tools is used by the web daemon: the *ESearch* component. ESearch searches and retrieves primary *IDs.* Each Entrez database refers to the data records within it by an integer ID. This procedure accepts two arguments: a "database", which is the SRA, and a "search strategy", which can be one or a set of terms and phrases with or without boolean operator (for example "illumina+AND+breast"). The output is a list of ID experiments correlated with the search parameters. Given one or more IDs, these are retrieves in text format from the Entrez search engine directly. In this phase the data are not downloaded: only the information from the samples (tissue, platform, study, ecc.) are acquired.

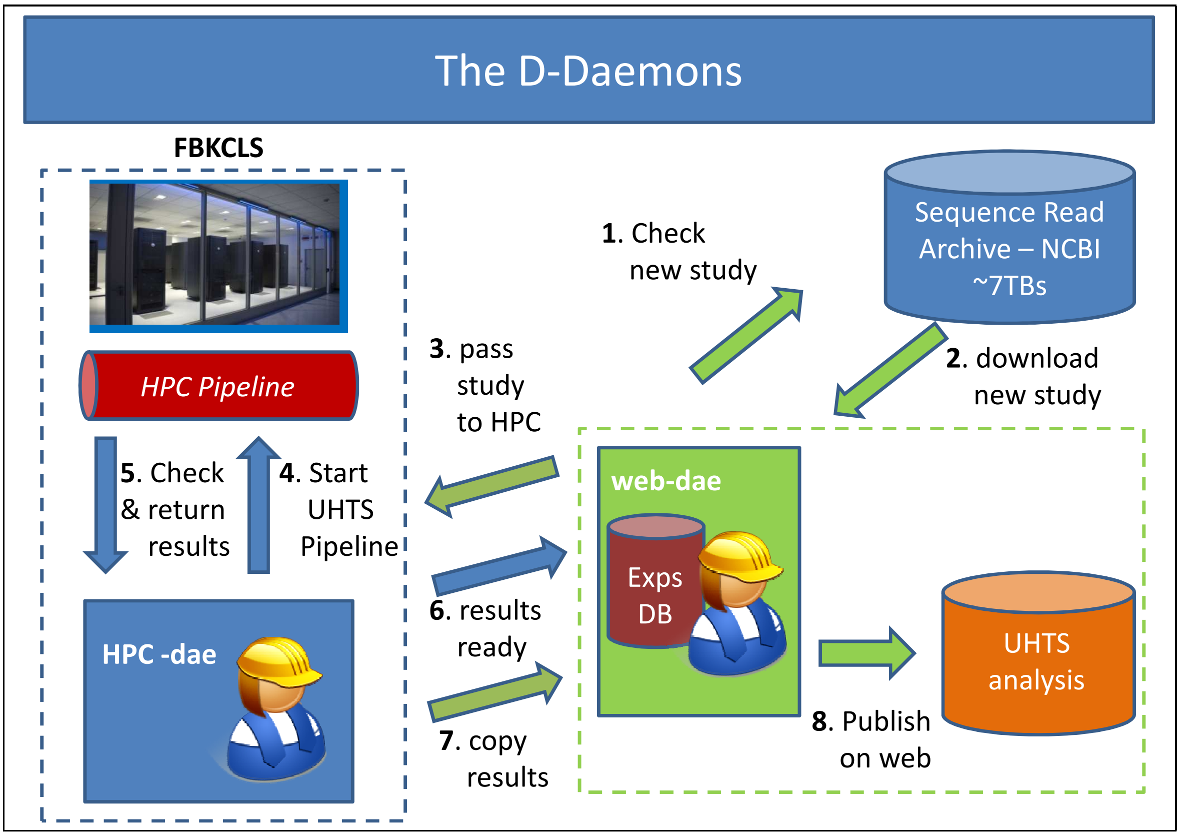


Figure 3.2: D-Daemons architecture.

The Entrez cross-database search and the ESearch utility are *common gateway interface (CGI)* scripts invoked from a uniform resource locator (URL) of a web server; their task is to analyze the content of the requests from clients (users through their web browsers), to determine an appropriate document to send in response, and to return it to the clients. Therefore, the web daemon utilizes Entrez and ESearch to perform a set of queries to their URLs. For example, given a text search strings “T”, a first query is performed to obtain the number of entries which match T. This is the URL for this step:

```
http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi
?db=sra&term=T&rettype=count
```

Parsing the output of the previous query, the number of entries, which will be indicated as “totexps”, is archived. Another web query is submitted to obtain the IDs of the corresponding entries:

```
http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi
?db=sra&term=$1&retmax=$totexps
```

Finally, once captured the IDs list, indicated as “L”, the documentations of each experiment is acquired through a loop that queries the Entrez search engine for each element in the catalog L:

```
while ID in L:
do
  http://www.ncbi.nlm.nih.gov/sites/entrez?cmd=retrieve
  &db=sra&list_uids=ID&dopt=full&format=text
done
```

As previous reported, only the notes about an experiment are stored in the SQLite database, but this information is sufficient to distinct each sample and to find their location in the SRA site. These records are listed by the web daemon to identify new items in the NCBI archive.

The second service is called the *HPC daemon (HPC-dae)* and it runs on the HPC infrastructure, where it can interact with a queue system. The HPC-dae is the software which coordinates how user programs (jobs) are scheduled to be executed on the execution nodes. The service waits for the sequence data from its counterpart, which has downloaded them from the SRA. The HPC service passes the experiments to the HPC pipeline, which processes them, checks when the analysis line is terminated and returns the output to the web daemon. The HPC service can monitor several HPC pipeline instances simultaneously and if this is considered with the fact that each pipeline is executed on several nodes concurrently, these attributes allow processing a huge number of ultra high throughput samples.

The web and HPC services talk each other through a *network socket*, which is an application programming interface (API) used to transmit data on TCP/IP based networks. Computer networks use this protocol to transmit information between machines. The TCP/IP defines a set communication channels called *ports*, identified by a number, where applications and services can send, receive and listen data. For example, the port 80, indicated as the HTTP port, is used by web browsers. For security reasons a restricted set of TCP/IP port can be used only in a local area network. Usually, the web servers and the HPC infrastructures are located in separated networks checked by a firewall element, which monitors and permits only communications on authorized ports. In the proposed system, the TCP/IP port of the socket between services is user-defined, therefore the suggested architecture can be installed in firewall network topologies easily. Finally, the network socket is also used in transmissions when the daemons run on the same workstation.

3.4 The HPC pipeline

The analysis pipe that processes the short read sequences in a UHTS experiment is composed by three tools: Bowtie, Samtools and Cufflinks, already presented in Chapter 2. They represent the state of the art in software tools applied to the next generations sequencing analysis. Bowtie is used to align the short reads on the reference genome. In the next chapter, a comparative analysis between this tool and another aligner is presented; the results motivate why Bowtie was chosen as the alignment engine of this pipeline. SAMtools transforms the Bowtie outputs in *bedGraph Track Format* [35] format files. The bedGraph type is extremely useful for visualizing probability scores and transcriptome data in the UCSC Genome Browser. The bedGraph is a line-oriented format and its four required fields are:

1. **chrom**: the name of the chromosome (e.g. chr3, chrY, chr5_random)

2. **chromStart**: the starting position of the feature in the chromosome

3. **chromEnd**: the ending position of the feature in the chromosome

4. **data value**: can be integer or real, positive or negative values.

Once these files are uploaded in the UCSC Genome Browser, the aligned short reads are visualized as tracks over the chromosomes of a reference genome. A binary version of the bedGraph file is also produced. This version utilizes the *bigWig* [36] file format and it is used to upload very large data in the UCSC Genome Browser. Therefore, through a bigWig file it is possible to visualize a huge number of alignments on an entire chromosome. The conversion between bedGraph and bigWig is archived by a utility script called “bedGraphToBigWig”, downloadable from the UCSC Genome Browser site directly. Within HPC-dae, the bedGraphToBigWig converter is called in HPC pipeline after the generation of bedGraph files.

SAMtools is also used to prepare the right input format to last module of the chain automatically. Cufflinks performs RPKM computation of the aligned sequences on genes. The RPKM measure [9] reflects the molar concentration of a transcript in the starting sample by normalizing for RNA length and for the total read number in the measurement. This measure is becoming a standard in next generation experiments, because it represents an objective measure allowing the comparison of the transcript levels in different experiments. The Cufflinks statistical model (described in Chapter 2, based on the methods in [15]) probabilistically assigns reads to individual isoforms: this process can identify new isoforms, which are reported in the result files.

All these software elements are not written to run in a HPC environment natively. Therefore, some adaptations have been applied to make these tools running in a efficient mode. In the Bowtie case, all the short reads that compose an experiment are divided into a number of groups given by the number of allocable execution nodes (it could be a workstation, a CPU or single core in a multi-core processor, depending on the HPC architecture). Each group of reads is assigned to one execution node, where one Bowtie instance performs the alignment on the reference genome: in this way each group of reads is processed in parallel. A different approach is followed to parallelize SAMtools and Cufflinks. The aligned short reads are divided on the basis of the chromosome of the reference genome onto which they where aligned. [1] For each chromosome an instance of the two tools is executed to generate the bedGraph and bigWig files, the RPKM values and a little utility that publishes outcome to create the HTML pages to publish these outcomes. It is possible to understand because the results

[1] For the UCSC HG18, used in this work, the number of chromosomes is 49. This number contains the standard 22, X and Y chromosomes, while others are virtual chromosomes and hold the Mitochondrial DNA and the alternative haplotype sequences, i.e. groups of alleles of different genes, of the real chromosomes.

are presented separated by chromosome. These new functionalities in the D-Daemons architecture to execute Bowtie, SAMtools and Cufflinks in a HPC environment are added by ad hoc shell scripts called "wrappers", which set the stage for running the software pipeline on several execution machines simultaneously.

3.5 SRA structure

The system queries the Sequence Read Archive (SRA) to find experiments on the basis of a search key defined by the user. In case of a positive match, the required samples are downloaded to be analyzed. Understand how these steps are performed, a survey on the SRA's structure has been necessary. In the SRA archive the next generation data are classified through a hierarchy. The organization is based on the following categories:

1. studies
2. experiments
3. samples
4. runs.

A *study* is a biological investigation where one ultra high throughput sequencing technology has been applied as analytic method. Most of these investigations are linked to a publication on a biology o bioinformatics journals. One study may be comprised of several experiments. An *experiment* reports specifically what was sequenced and the procedure followed used. It includes information about the source of the DNA, the sample, the sequencing platform, and the processing of the data. The modern UHTS platforms utilize procedures where a sample is sequenced through a series of parallel pipelines, which are called "lines" or "runs". In the SRA a *run* contains the short reads from each platform run. Each experiment is composed of one or more sequencing it. In the future, some studies will also have an associated analysis. These “analyses” may include assemblies of the short reads into genomic or transcript contigs, i.e. long contiguous DNA sequences assembled from short sequencing (short reads), and alignment to existing genomes or alignments onto other SRA data. Records from each class have unique accession identifiers with a specific three-letter prefix that indicates the type:

ERP or SRP for studies (example: SRP000727)

SRS for samples (example: SRS002671)

SRX for experiments (example: SRX003932)

SRR for runs (examples: SRR015321, SRR015322, SRR015323).

As mentioned in Chapter 2, the data in the SRA archive can be found through Entrez cross-database search. The Entrez search is available directly from the SRA web portal [10] and it makes possible to select experiments that match search strings entered by a user. For each entry returned by a query a lot of information is returned. In general the following details are produced for each experiment in the NCBI repository:

Accession:	SRX003932
Title:	
Experiment Design:	
Submission:	GEO
Study accession:	SRP000727
Study Title:	Alternative Isoform Regulation in Human Tissue Transcriptomes

Study Abstract:	
Study Center:	GEO
Study Center Project:	Alternative Isoform Regulation in Human Tissue Transcriptomes (ID=0)
Project name:	Alternative Isoform Regulation in Human Tissue Transcriptomes
Sample Accession:	SRS002671
Sample Description: source:	Human adipose tissue; description: RNA_seq
Sample Common Name:	Homo sapiens, (TaxonId=9606)
Sample Attributes:	tissue=adipose
Sample Links:	GEO web Link= http://www.ncbi.nlm.nih.gov /geo/query/acc.cgi?acc=GSM325481
Library Name:	adipose
Library Strategy:	EST
Library Source:	NON GENOMIC
Library Selection:	cDNA
Library Layout:	SINGLE
Platform Name:	ILLUMINA
Processing:	Base Space,
Quality score:	, 0x0.0E0
Spot descriptor:	1) Application Read, Forward
Total:	7 runs, 27.8M spots, 888.1M bases
Run #1:	SRR015321, 3851492 spots, 123247744 bases
Run #2:	SRR015322, 3879376 spots, 124140032 bases
Run #3:	·········

The experiment's unique accession identifier is also used to organize files in the SRA's FTP site. The root directory of this ftp repository is

`ftp://ftp.ncbi.nlm.nih.gov/sra/static/`

The files of each experiment are stored in directories under this ftp root and these subdirectories have a name which is assembled by the experiment identifier. The subdirectory path is composed by the first six letter of the experiment identifier, the slash symbol and the complete experiment identifier. For example the experiment called SRX003932 has its runs file in the following sub directory:

SRX003/SRX003932

while the full path becomes:

`ftp://ftp.ncbi.nlm.nih.gov/sra/static/SRX003/SRX003932/`

This relatively simple encoding defines the structure of a huge amount of biological knowledge, as 11 Terabases of sequencing data are included in the Sequence Read Archive repository.

3.6 Integration in Galaxy

The Galaxy framework has been introduced in Chapter 2. It is an environment with an easy-to-use web interface where different software for UHTS studies can be integrated and executed individually. Alternatively, the software tools can be concatenated and run Each experiment is composed of one or more sequencing it. together to combine a sequence of analysis steps called "workflow".

One characteristic of Galaxy is that it supports Perl and Python scripts. The proposed web-dae service is also written in Python language, therefore a Galaxy framework resulted an ideal candidate for a user interface.

A set of D-Daemons tools has been developed to be included in the tools panel of the Galaxy interface. These new elements are python scripts which call the functions of web-dae component. Under a menu tool called *D-Daemons tools* (see Figure 3.3) seven new entries are developed:

1. **NCBI-SRA to SQLite DB** This command queries the SRA repository by a user defined research key and saves the result in a SQLite database.

2. **SQLite DB loader** Given an existing database (produced in a previously query) this entry resubmits the user defined research key (saved in the DB) and if new experiments are found they are saved in the database.

3. **NCBI-SRA Update to SQLite DB** Given an existing database (produced in a previously query) this entry resubmits the user defined research key (saved in the DB) and if new experiments are found they are saved in the database.

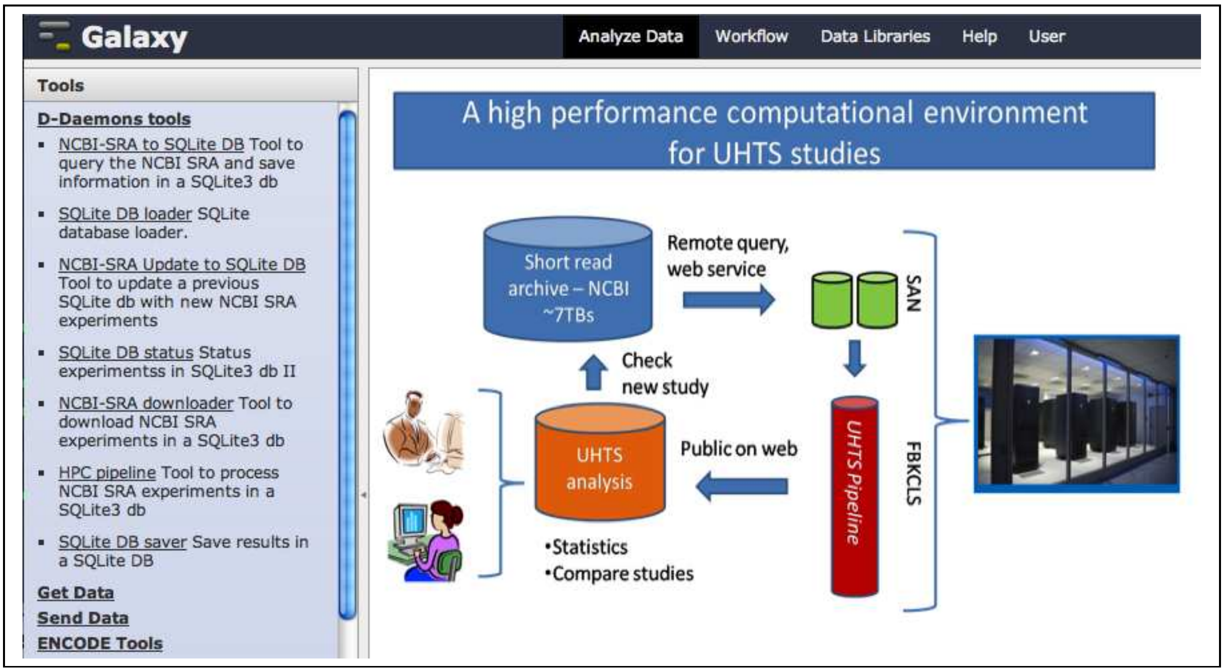
4. **SQLite DB status** This command prints the status of experiments saved in a DB. This view presents experiments by their SRA codes, file directories and result links.

5. **NCBI-SRA downloader** This function downloads files of the experiments in the DB..

6. **HPC pipeline** This function takes as input an user-selected experiment from a database and communicates with the HPC daemon to be processed in the HPC pipeline. Moreover, it is possible to analyze all non yet processed experiments in a database.

7. **SQLite DB saver** This function saves the SQLite DB in the Galaxy history to an external file.

These SRA modules are based on some features of the web daemon and they can be used a separately. A main feature in Galaxy is to build and execute workflows of different tools. Thus, the presented Galaxy-like modules of the D-Daemons system can be concatenated for new analysis workflows (Figure 3.4).



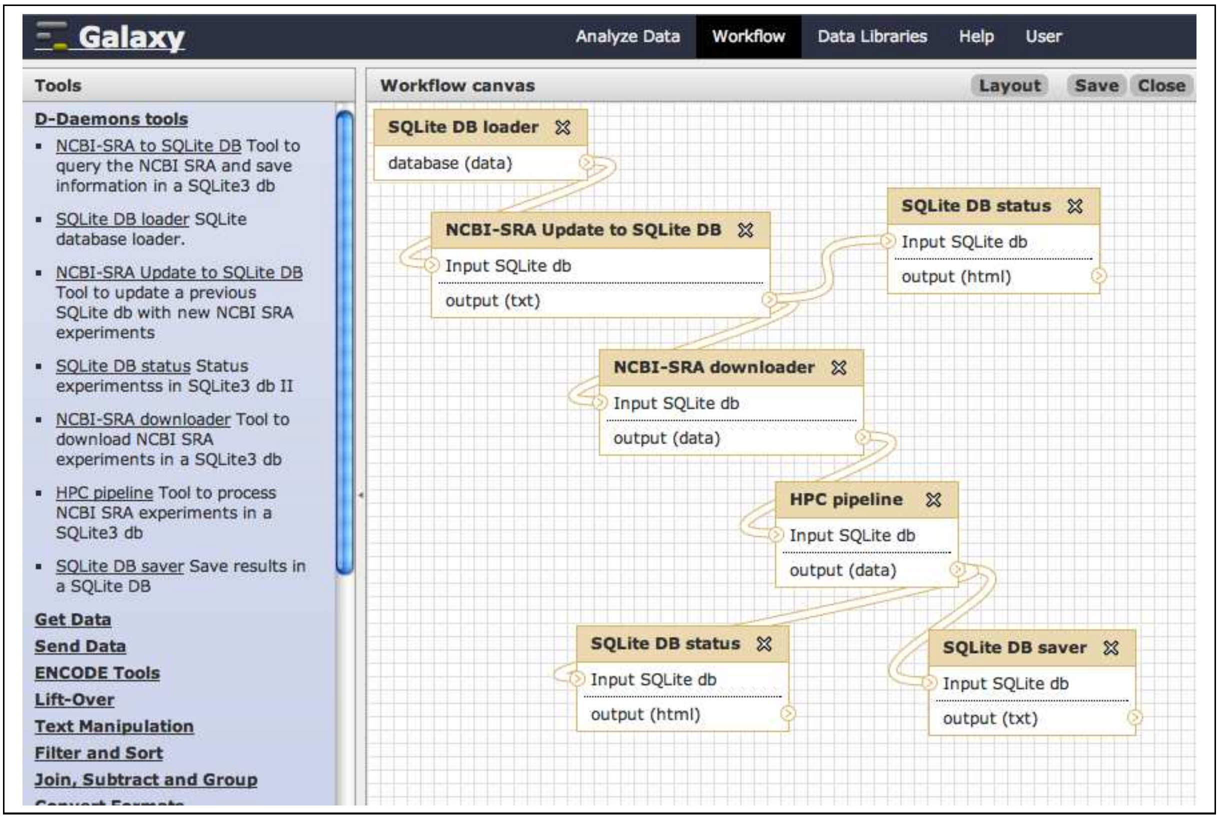
D-Daemons tools

- NCBI-SRA to SQLite DB Tool to query the NCBI SRA and save information in a SQLite3 db
- SQLite DB loader SQLite database loader.
- NCBI-SRA Update to SQLite DB Tool to update a previous SQLite db with new NCBI SRA experiments
- SQLite DB status Status experimentss in SQLite3 db II
- NCBI-SRA downloader Tool to download NCBI SRA experiments in a SQLite3 db
- HPC pipeline Tool to process NCBI SRA experiments in a SQLite3 db
- SQLite DB saver Save results in a SQLite DB

Get Data
Send Data
ENCODE Tools

A high performance computational environment for UHTS studies
Short read archive – NCBI ~7TBs
Remote query, web service
SAN
Check new study
UHTS analysis
Public on web
UHTS Pipeline
FBKCLS
•Statistics
•Compare studies

Figure 3.3: The Galaxy main page and on the left the toolbar with the D-Daemons menu.



D-Daemons tools

- NCBI-SRA to SQLite DB Tool to query the NCBI SRA and save information in a SQLite3 db
- SQLite DB loader SQLite database loader.
- NCBI-SRA Update to SQLite DB Tool to update a previous SQLite db with new NCBI SRA experiments
- SQLite DB status Status experimentss in SQLite3 db II
- NCBI-SRA downloader Tool to download NCBI SRA experiments in a SQLite3 db
- HPC pipeline Tool to process NCBI SRA experiments in a SQLite3 db
- SQLite DB saver Save results in a SQLite DB

Get Data
Send Data
ENCODE Tools
Lift-Over
Text Manipulation
Filter and Sort
Join, Subtract and Group

Figure 3.4: A workflow based on elements of the D-Daemons architecture

3.7 Test environment

The HPC pipeline has been developed in a Linux cluster environment composed by 416 cores in 52 nodes, with a queue system based on the Sun Grid Engine software [37] and a 10 Terabyte volume in a storage area network to stock the input and output data. Different Linux workstations outside the Linux cluster network have been used to implement the dual services architecture between the cluster front end and a web server.

3.8 The CRC case study

The D-Daemons architecture has been tested on different analysis tasks. Here a typical case study on cancer UHTS data is presented in details. In the context of the EU FP7 project HIPERDART, in collaboration with Istituto Catalao de Oncologia, a FBK [38] researcher needs to recover all Colorectal Cancer Cell (CRC) lines in SRA. Through "Colorectal Cancer cell line" query, all experiments stored in NCBI SRA were downloaded and processed by the system. The Experiment features are shown in the following table.

SRA experiment code	SRA sample code	Runs	Spots	Bases
SRX012945	SRS007122	2	19M	685.5M
SRX012946	SRS007123	2	18.3M	659.4M

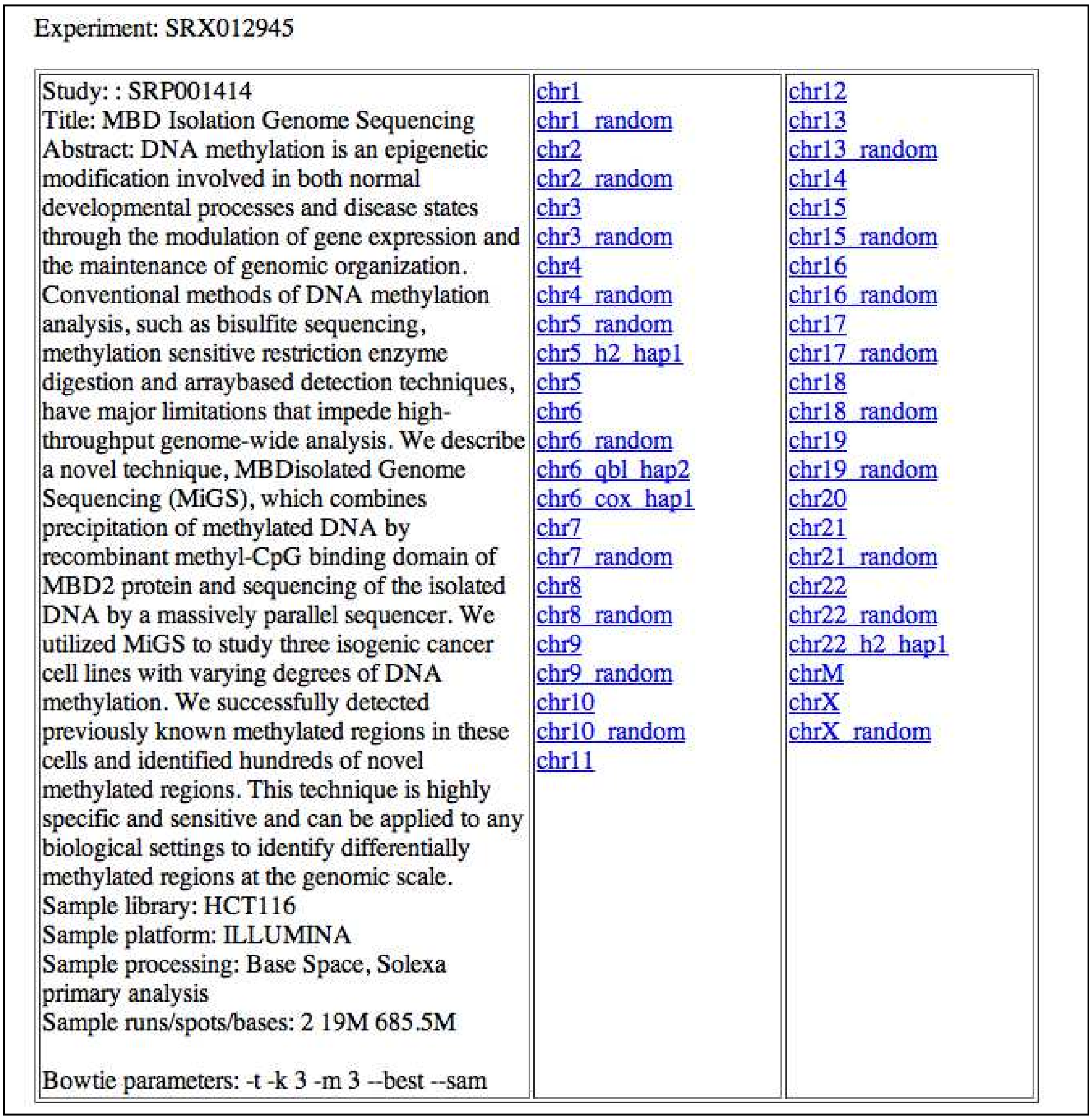
What follows is an example of the output generated by the D-Daemons system. The HTML pages where experiment results are published (in Figure 3.5 and Figure 3.6). The output is divided on the HG18 chromosomes. In Figure 3.7 and Figure 3.8 outputs for each chromosome:

1. RPKM measure

2. bedGraph file
3. bigWig file
4. Transcripts gtf (from Cufflinks)
5. Transcripts tmap (from Cufflinks)
6. UCSC Genome Browser track line (to load bigWig file in UCSC Genome Browser)
7. link to open wigBig in the UCSC Genome Browser.

Figure 3.9 and Figure 3.10 present the RPKM values sorted from high numbers to low numbers. The second column, produced by the Cufflinks software, allows to identify possible “new junctions”.

Finally, in Figure 3.11 and Figure 3.12 aligned short reads (on HG18 chromosome 1) of the Colorectal Cancer cell line experiments are visualized in the UCSC Genome Browser site [2]. Through the bigWig format [36], it is possible to display alignments on the entire chromosome length.



Experiment: SRX012945

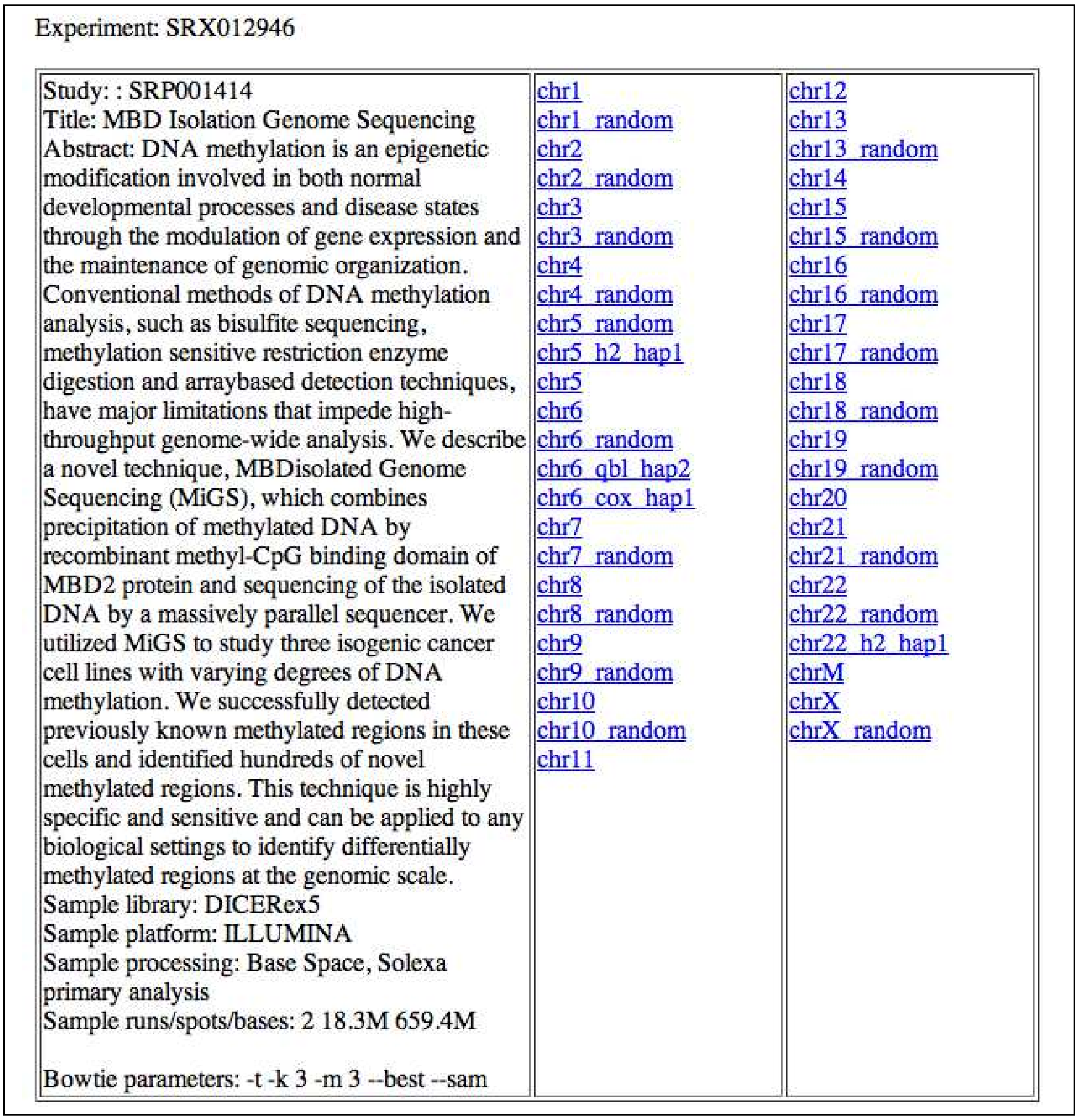
Study: : SRP001414
Title: MBD Isolation Genome Sequencing
Abstract: DNA methylation is an epigenetic modification involved in both normal developmental processes and disease states through the modulation of gene expression and the maintenance of genomic organization. Conventional methods of DNA methylation analysis, such as bisulfite sequencing, methylation sensitive restriction enzyme digestion and arraybased detection techniques, have major limitations that impede high-throughput genome-wide analysis. We describe a novel technique, MBDisolated Genome Sequencing (MiGS), which combines precipitation of methylated DNA by recombinant methyl-CpG binding domain of MBD2 protein and sequencing of the isolated DNA by a massively parallel sequencer. We utilized MiGS to study three isogenic cancer cell lines with varying degrees of DNA methylation. We successfully detected previously known methylated regions in these cells and identified hundreds of novel methylated regions. This technique is highly specific and sensitive and can be applied to any biological settings to identify differentially methylated regions at the genomic scale.
Sample library: HCT116
Sample platform: ILLUMINA
Sample processing: Base Space, Solexa primary analysis
Sample runs/spots/bases: 2 19M 685.5M

Bowtie parameters: -t -k 3 -m 3 --best --sam

chr1
chr1_random
chr2
chr2_random
chr3
chr3_random
chr4
chr4_random
chr5_random
chr5_h2_hap1
chr5
chr6
chr6_random
chr6_qbl_hap2
chr6_cox_hap1
chr7
chr7_random
chr8
chr8_random
chr9
chr9_random
chr10
chr10_random
chr11

chr12
chr13
chr13_random
chr14
chr15
chr15_random
chr16
chr16_random
chr17
chr17_random
chr18
chr18_random
chr19
chr19_random
chr20
chr21
chr21_random
chr22
chr22_random
chr22_h2_hap1
chrM
chrX
chrX_random

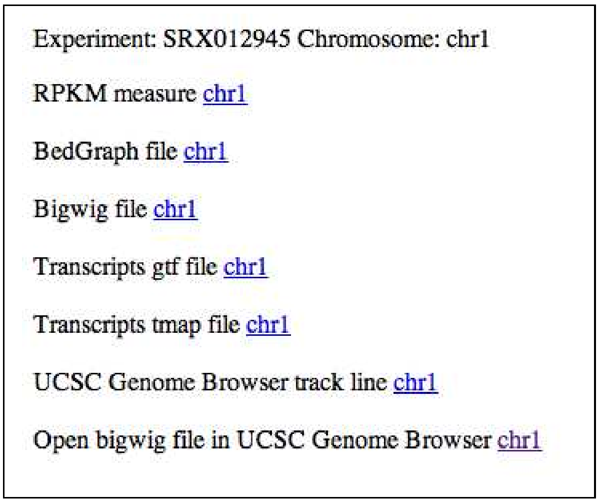
Figure 3.5: Main result page for experiment SRX012945.



Experiment: SRX012946

Study: : SRP001414 Title: MBD Isolation Genome Sequencing Abstract: DNA methylation is an epigenetic modification involved in both normal developmental processes and disease states through the modulation of gene expression and the maintenance of genomic organization. Conventional methods of DNA methylation analysis, such as bisulfite sequencing, methylation sensitive restriction enzyme digestion and arraybased detection techniques, have major limitations that impede high-throughput genome-wide analysis. We describe a novel technique, MBDisolated Genome Sequencing (MiGS), which combines precipitation of methylated DNA by recombinant methyl-CpG binding domain of MBD2 protein and sequencing of the isolated DNA by a massively parallel sequencer. We utilized MiGS to study three isogenic cancer cell lines with varying degrees of DNA methylation. We successfully detected previously known methylated regions in these cells and identified hundreds of novel methylated regions. This technique is highly specific and sensitive and can be applied to any biological settings to identify differentially methylated regions at the genomic scale. Sample library: DICERex5 Sample platform: ILLUMINA Sample processing: Base Space, Solexa primary analysis Sample runs/spots/bases: 2 18.3M 659.4M Bowtie parameters: -t -k 3 -m 3 --best --sam	chr1 chr1_random chr2 chr2_random chr3 chr3_random chr4 chr4_random chr5_random chr5_h2_hap1 chr5 chr6 chr6_random chr6_qbl_hap2 chr6_cox_hap1 chr7 chr7_random chr8 chr8_random chr9 chr9_random chr10 chr10_random chr11	chr12 chr13 chr13_random chr14 chr15 chr15_random chr16 chr16_random chr17 chr17_random chr18 chr18_random chr19 chr19_random chr20 chr21 chr21_random chr22 chr22_random chr22_h2_hap1 chrM chrX chrX_random

Figure 3.6: Main result page for experiment SRX012946.



Experiment: SRX012945 Chromosome: chr1

RPKM measure chr1

BedGraph file chr1

Bigwig file chr1

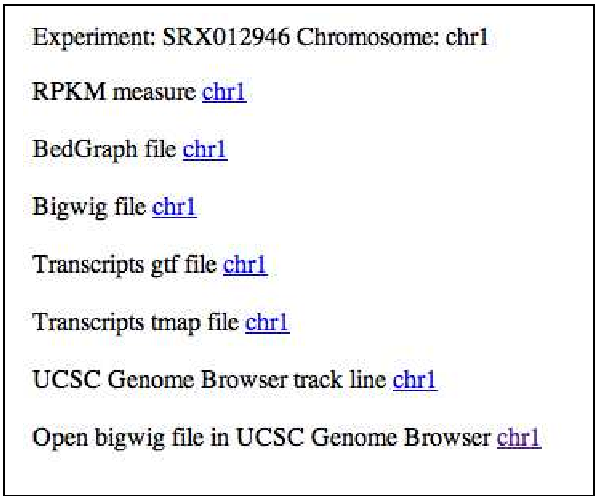
Transcripts gtf file chr1

Transcripts tmap file chr1

UCSC Genome Browser track line chr1

Open bigwig file in UCSC Genome Browser chr1

Figure 3.7: Chromosome 1 result page for experiment SRX012945.



Experiment: SRX012946 Chromosome: chr1

RPKM measure chr1

BedGraph file chr1

Bigwig file chr1

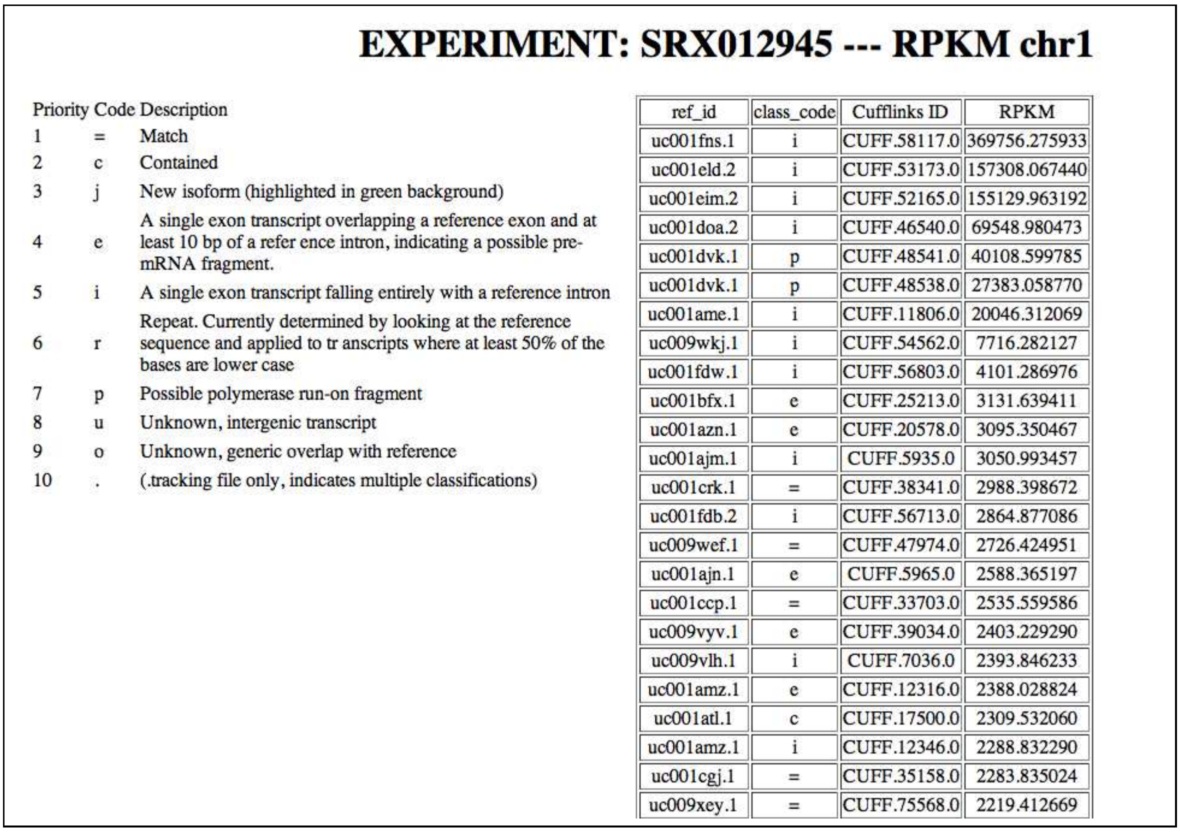
Transcripts gtf file chr1

Transcripts tmap file chr1

UCSC Genome Browser track line chr1

Open bigwig file in UCSC Genome Browser chr1

Figure 3.8: Chromosome 1 result page for experiment SRX012946.

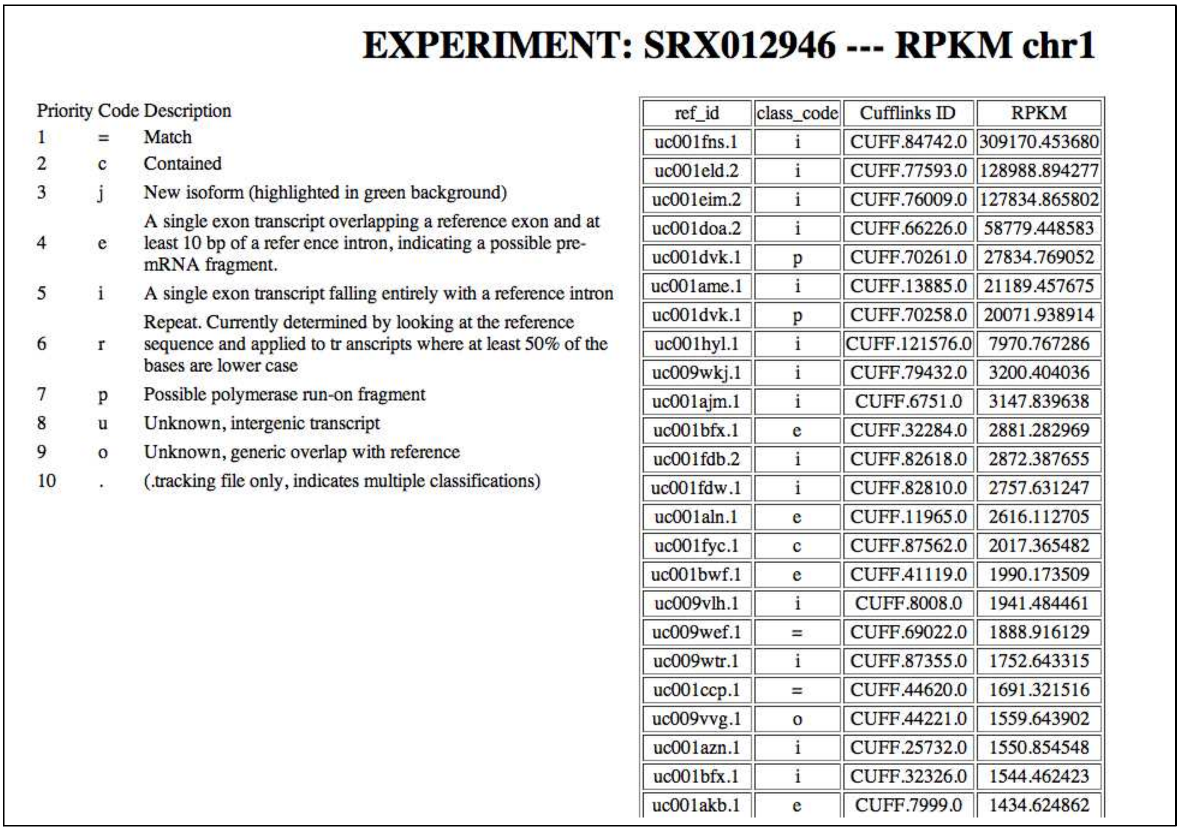


EXPERIMENT: SRX012945 --- RPKM chr1

Priority	Code	Description
1	=	Match
2	c	Contained
3	j	New isoform (highlighted in green background)
4	e	A single exon transcript overlapping a reference exon and at least 10 bp of a refer ence intron, indicating a possible pre-mRNA fragment.
5	i	A single exon transcript falling entirely with a reference intron
6	r	Repeat. Currently determined by looking at the reference sequence and applied to tr anscripts where at least 50% of the bases are lower case
7	p	Possible polymerase run-on fragment
8	u	Unknown, intergenic transcript
9	o	Unknown, generic overlap with reference
10	.	(.tracking file only, indicates multiple classifications)

ref_id	class_code	Cufflinks ID	RPKM
uc001fns.1	i	CUFF.58117.0	369756.275933
uc001eld.2	i	CUFF.53173.0	157308.067440
uc001eim.2	i	CUFF.52165.0	155129.963192
uc001doa.2	i	CUFF.46540.0	69548.980473
uc001dvk.1	p	CUFF.48541.0	40108.599785
uc001dvk.1	p	CUFF.48538.0	27383.058770
uc001ame.1	i	CUFF.11806.0	20046.312069
uc009wkj.1	i	CUFF.54562.0	7716.282127
uc001fdw.1	i	CUFF.56803.0	4101.286976
uc001bfx.1	e	CUFF.25213.0	3131.639411
uc001azn.1	e	CUFF.20578.0	3095.350467
uc001ajm.1	i	CUFF.5935.0	3050.993457
uc001crk.1	=	CUFF.38341.0	2988.398672
uc001fdb.2	i	CUFF.56713.0	2864.877086
uc009wef.1	=	CUFF.47974.0	2726.424951
uc001ajn.1	e	CUFF.5965.0	2588.365197
uc001ccp.1	=	CUFF.33703.0	2535.559586
uc009vyv.1	e	CUFF.39034.0	2403.229290
uc009vlh.1	i	CUFF.7036.0	2393.846233
uc001amz.1	e	CUFF.12316.0	2388.028824
uc001atl.1	c	CUFF.17500.0	2309.532060
uc001amz.1	i	CUFF.12346.0	2288.832290
uc001cgj.1	=	CUFF.35158.0	2283.835024
uc009xey.1	=	CUFF.75568.0	2219.412669

Figure 3.9: RPKM values for Chromosome 1 experiment SRX012945.

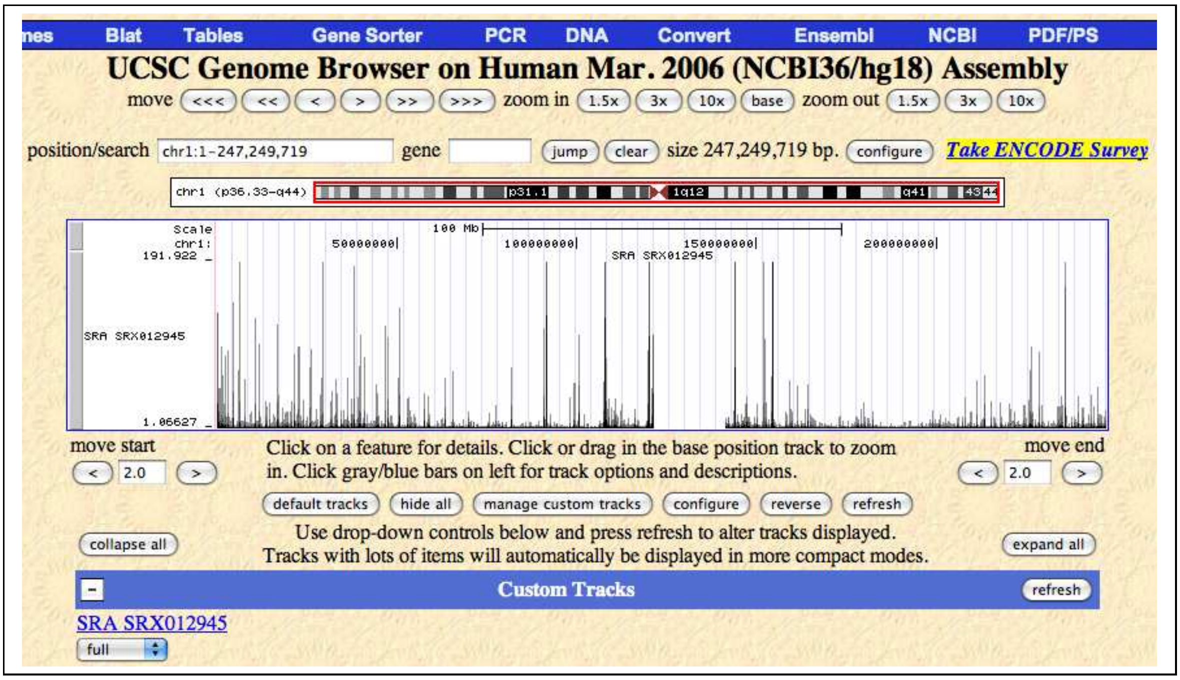


EXPERIMENT: SRX012946 --- RPKM chr1

Priority	Code	Description
1	=	Match
2	c	Contained
3	j	New isoform (highlighted in green background)
4	e	A single exon transcript overlapping a reference exon and at least 10 bp of a refer ence intron, indicating a possible pre-mRNA fragment.
5	i	A single exon transcript falling entirely with a reference intron
6	r	Repeat. Currently determined by looking at the reference sequence and applied to tr anscripts where at least 50% of the bases are lower case
7	p	Possible polymerase run-on fragment
8	u	Unknown, intergenic transcript
9	o	Unknown, generic overlap with reference
10	.	(.tracking file only, indicates multiple classifications)

ref_id	class_code	Cufflinks ID	RPKM
uc001fns.1	i	CUFF.84742.0	309170.453680
uc001eld.2	i	CUFF.77593.0	128988.894277
uc001eim.2	i	CUFF.76009.0	127834.865802
uc001doa.2	i	CUFF.66226.0	58779.448583
uc001dvk.1	p	CUFF.70261.0	27834.769052
uc001ame.1	i	CUFF.13885.0	21189.457675
uc001dvk.1	p	CUFF.70258.0	20071.938914
uc001hyl.1	i	CUFF.121576.0	7970.767286
uc009wkj.1	i	CUFF.79432.0	3200.404036
uc001ajm.1	i	CUFF.6751.0	3147.839638
uc001bfx.1	e	CUFF.32284.0	2881.282969
uc001fdb.2	i	CUFF.82618.0	2872.387655
uc001fdw.1	i	CUFF.82810.0	2757.631247
uc001aln.1	e	CUFF.11965.0	2616.112705
uc001fyc.1	c	CUFF.87562.0	2017.365482
uc001bwf.1	e	CUFF.41119.0	1990.173509
uc009vlh.1	i	CUFF.8008.0	1941.484461
uc009wef.1	=	CUFF.69022.0	1888.916129
uc009wtr.1	i	CUFF.87355.0	1752.643315
uc001ccp.1	=	CUFF.44620.0	1691.321516
uc009vvg.1	o	CUFF.44221.0	1559.643902
uc001azn.1	i	CUFF.25732.0	1550.854548
uc001bfx.1	i	CUFF.32326.0	1544.462423
uc001akb.1	e	CUFF.7999.0	1434.624862

Figure 3.10: RPKM values for Chromosome 1 experiment SRX012946.



UCSC Genome Browser on Human Mar. 2006 (NCBI36/hg18) Assembly

position/search chr1:1-247,249,719

size 247,249,719 bp.

move start 2.0

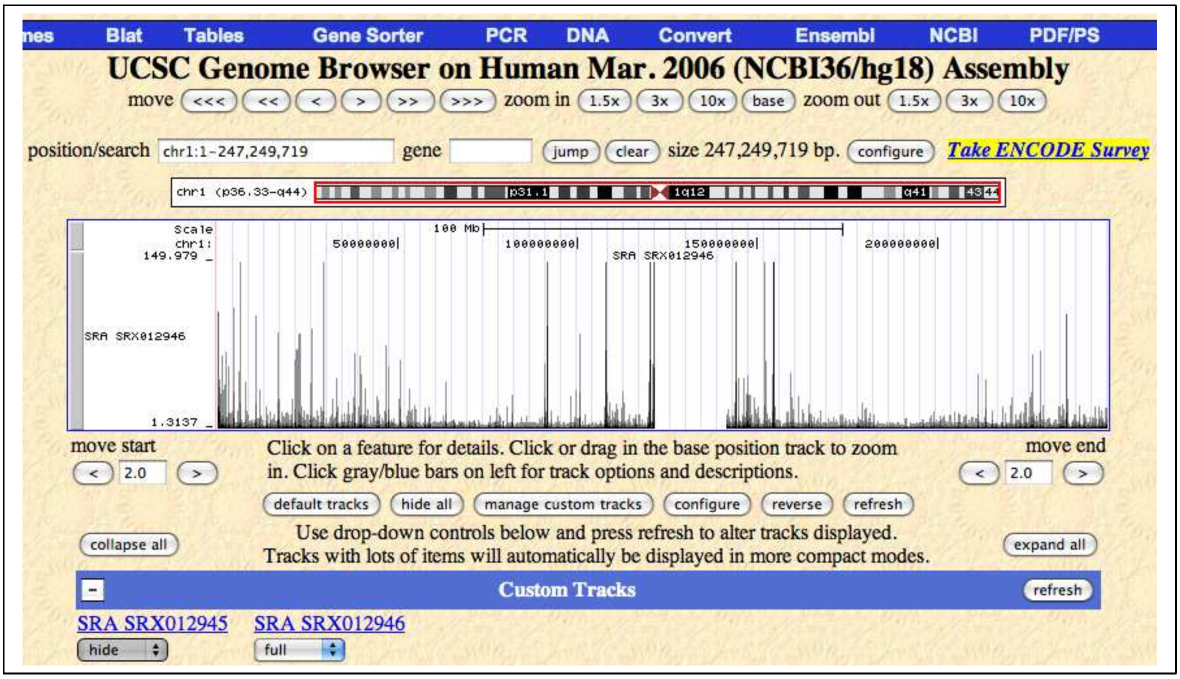
move end 2.0

Click on a feature for details. Click or drag in the base position track to zoom in. Click gray/blue bars on left for track options and descriptions.

Use drop-down controls below and press refresh to alter tracks displayed.
Tracks with lots of items will automatically be displayed in more compact modes.

SRA SRX012945: full

Figure 3.11: Alignments on chromosome 1 for experiment SRX012945 visualized in the UCSC Genome Browser.



position/search chr1:1-247,249,719 gene size 247,249,719 bp.

Click on a feature for details. Click or drag in the base position track to zoom in. Click gray/blue bars on left for track options and descriptions.

Use drop-down controls below and press refresh to alter tracks displayed. Tracks with lots of items will automatically be displayed in more compact modes.

SRA SRX012945: hide

SRA SRX012946: full

Figure 3.12: Alignments on chromosome 1 for experiment SRX012946 visualized in the UCSC Genome Browser.

Chapter 4

Aligner selection in a NGS pipeline

4.1 Aligner comparison

This section describes the benchmarks, on a SNP discovery from RNA-seq data, obtained with two recent Open Source Software currently popular in next generation sequencing (NGS) experiments: Bowtie [3] and Bwa [6]. This work aims to compare two candidate alignment engines. The winner will be inserted in the proposed HPC pipeline presented in the Chapter 3. Moreover this exercise exemplifies how to control reproducibility in NGS experiments in complex pipelines. The two aligners were tested on three RNA-Seq datasets previously analyzed in the Wang study [1]. Data are available from the NCBI Sequence Read Archive [10, 26] with accession SRA002355.1. The dataset is formed by 32bp short reads produced by the Illumina sequencing process of a Human Breast tissue and two cell lines (BT474 & MCF7). Following are other attributes of the samples:

SRA signature	Tissue/Cell Line	Reads	Lanes
SRX003922	BREAST	16120746	4
SRX003923	MCF7	16059515	4
SRX003935	BT747	18424533	7

The human build HG18 from the UCSC Genome Bioinformatics site [20] was used as reference genome. Our trials are focused to test the behaviors of the two aligners in a parallel environment, where they can be executed on several CPUs simultaneously and the main intention is to reduce the alignment time. Here are compared time, memory usage and alignment accuracy of Bowtie and Bwa executed in parallel mode (i.e. on more than one CPUs).

The accuracy measures are performed taking in account the number of uniquely mapped reads and of allowed mismatches and through a SNP discovery analysis. Bowtie and Bwa do not include functions for this type of analysis, so other two instruments have been used on their outputs to perform the SNP detection: TopHat [8] has been applied on Bowtie output, while SAMTool [11] has been run on the Bwa results.

The FBK [38] computing system has been used to perform these trial estimations. The facility is composed by 416 cores in 52 nodes, each one with 2 Intel Xeon Quadcore processors and RAM Memory ranging from 16 to 72 GB per nodes. The operating system installed on nodes is the Scientific Linux distribution, which is developed by CERN and Fermi Lab. A SGE [37] queue system governs users and their programs (called jobs) on the cluster. The queue system schedules users' jobs on available free nodes to start their execution. This HPC infrastructure has allowed us to complete tests using a number of processor cores from 1 to 256. Current processors are based on multi-core architecture, where two or more CPUs (called cores) have been condensed in the same die for increased performance. Given this clarification, in the following the term "CPU" will be used to indicate a "processor core".

The two aligners are written to run on a single workstation and at best they support to be forked in some threads always on a single machine. Multithreading is not enough to run and operate efficiently in a parallel environment as a computer cluster. Modern cluster facilities are based on distributed memory architectures, where each node views only its local Random-access memory (RAM). Multiple threads can be applied in local RAM memory space, so in a cluster environment multithread software can use only one node at time.

Given these boundaries, to execute Bowtie and Bwa on two o more nodes up to 256 cores at the same time, two ad hoc shell scripts (or "wrappers") were developed to make the aligners more compliant with the cluster's queue system. These wrappers allow Bowtie and Bwa to interact and run in the cluster facility, implementing two methods to distribute the input files/reads to multiple instances of the two aligners running on different CPUs. How to allocate various occurrences of an aligner, share out the input reads between them and rebuild the final output are the keystones to implement the *parallelization* of Bowtie and Bwa. Two parallelization strategies are put into action through the two wrappers.

The first method is based on some attributes of the Illumina sequencing process. Here, the short reads are divided in $1-7$ files, called *lanes*. These files reflect the architecture of the Illumina system, where the sequencing process runs over 7 pipelines (+1 for control). So, the tactic is to submit a number of instances of one aligner which is equal to the number of lanes, i.e. each single lane is assigned to one instance of Bowtie or Bwa to be aligned. When computing of all instances is finished, their output is unified in a single output file, which contains the alignment result of all short reads from all lanes. This wrapper will be indicated as the *separated lanes method*.

The second method does not consider the lane separation and it is based on the number N of CPUs which one user wants to utilize. The procedure considers all T reads from all the lanes, it divides them in N groups, where each one contains an uniform quantity of short reads, $[T/N]$ (because in general N is not a factor of T). These N groups are processed by N copies of an aligner. As in the previous methods, when all instances have finished, all the output files are combined in a unique file. This routine will be marked as the *merged lanes method*.

The behaviors of the two aligners and the parallelization methods have been tested recording the RAM memory usage and the computational time to execute an alignment of the short reads from the BREAST, MCF7 and BT747 samples. All trials involved in profile collation of performance in time and memory RAM usage have been repeated four times; means and standard deviations of these repetitions for each specimen are reported in tables below.

Besides the computing performances quality of the results from the two aligners were also inspected. Uniquely mapped reads and allowed mismatches have been quantified and SNP discovery has been applied on the alignment files from the BREAST, MCF7 and BT747 samples. Consensus building and downstream analysis was performed by SAMtools [11], a set of utilities for the manipulation of mapped reads. SNPs were called from the consensus sequence and then filtered in order to retain only high-quality SNPs, based on the following inclusion criteria:

- Mapping quality > 25
- Read depth > 3
- SNPs do not fall within 10bp from a gap.

As reported above, the Bowtie output has to be used through Tophat to produce a SNP output, so the mark “Bowtie/T” will denote the use of both for this aim.

The graphs in Figure 4.1, Figure 4.3 and Figure 4.5 show times in seconds spent by Bowtie and Bwa (curves in blue and red color respectively) with the merged lanes method varying the number of uses CPUs applied to the three samples. Table 4.1, Table 4.3 and Table 4.5 report values of these test conditions.

The graphs in Figure 4.2, Figure 4.4 and Figure 4.6 display times in seconds spent by Bowtie and Bwa with the separated lanes method varying the number of used CPUs applied to the lanes of Breast, BT474 and MCF7 samples. The column lane indicates the number of CPUs in charge of aligning all lanes, for example the 3 value denotes that all lanes are equally distributed and aligned on 3 CPUs. Table 4.2, Table 4.4 and Table 4.6 report precise numbers.

In Figure 4.7, Figure 4.8 and Figure 4.9 the progress of used RAM memory, expressed in Gigabyte (GB), by Bowtie and Bwa is shown changing the involved CPUs number in the merged lanes method. In Table 4.7, Table4.9 and Table 4.11 the corresponding numerical values are shown.

The separated lanes method and its RAM memory consumption is accurately reported in Table 4.8, Table 4.10 and Table 4.12. Given the constant values obtained, no graphs have been supplied for this method.

Counts of uniquely mapped reads, i.e., reads mapped exactly on the reference genome, have been produced. Moreover, amounts of mismatches reads, i.e., reads aligned on the reference genome with some errors, have been computed considering 3 cases: 1 mistake, 2 mistakes, or 3 and more mistakes. These values are presented in Figure 4.10 and Table 4.13 for each aligner and sample. The SNP analysis is reported in the distribution of single base mismatches in merged and separated lanes cases for the Breast data in Figure 4.11, Figure 4.12, Figure 4.12 and Figure 4.13. As a first example of application, in Figure 4.15 and Figure 4.16 the coverage of aligned short reads produced by Bowtie and Bwa are focused on three specific genes: SPARC, ATOX1 and G3BP1.

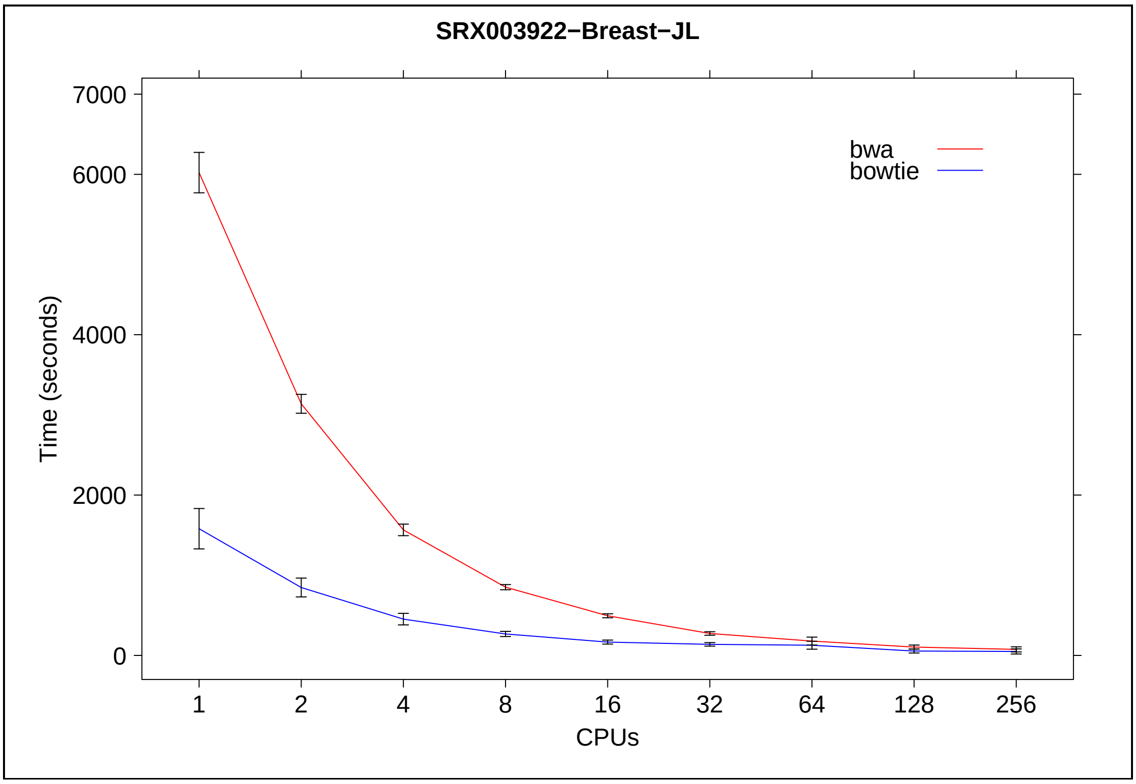


Figure 4.1: Scalability comparison of Bowtie and BWA on merged lanes of Breast sample.

	BWA		BOWTIE	
CPU	mean	sd	mean	sd
1	6021.75	251.97	1580.25	57.59
2	3138.38	117.48	847.12	35.63
4	1565.69	72.11	452.75	23.12
8	851.56	32.20	267.91	25.53
16	494.70	24.80	166.53	28.73
32	273.84	21.94	138.16	76.08
64	178.96	49.25	127.11	64.89
128	104.39	24.88	54.79	20.35
256	75.35	31.87	49.58	24.20

Table 4.1: Scalability (in seconds) on Breast sample with merged lanes.

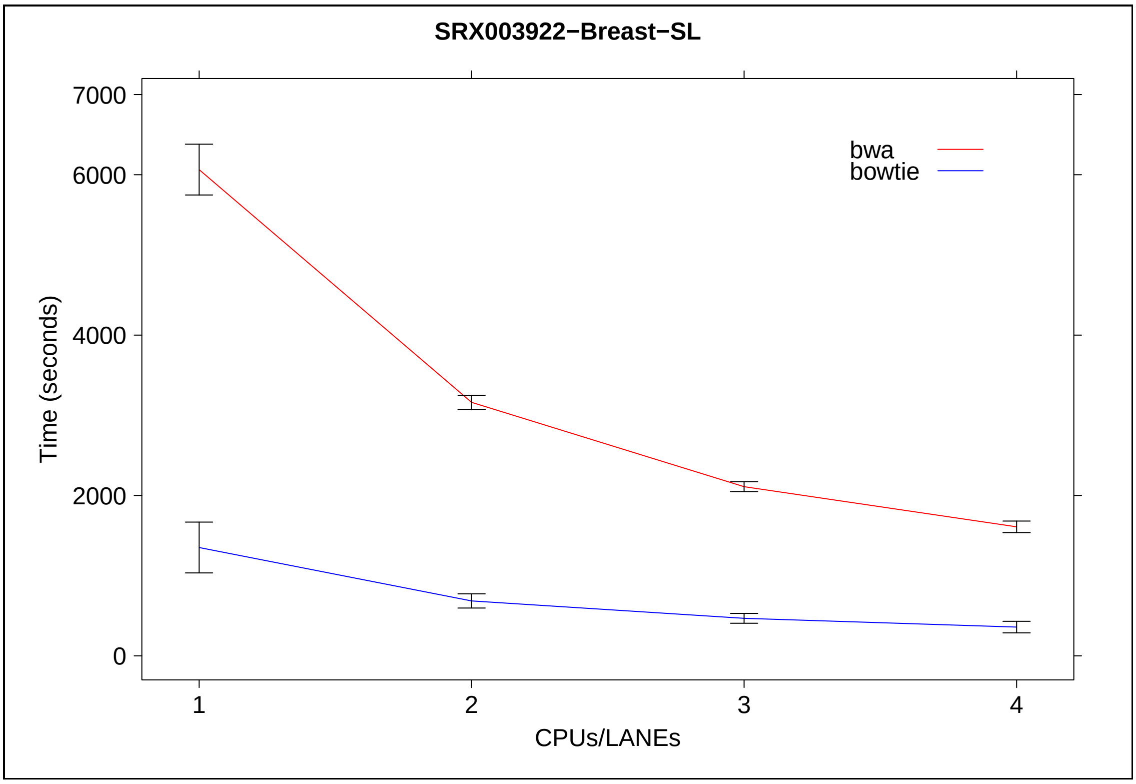


Figure 4.2: Scalability comparison of Bowtie and BWA on separated lanes of Breast sample.

	BWA		BOWTIE	
lane	mean	sd	mean	sd
1	6064.75	316.95	1351.00	49.89
2	3161.62	88.46	684.75	18.42
3	2109.75	61.49	467.75	27.49
4	1609.06	71.88	358.62	22.00

Table 4.2: Scalability (in seconds) on Breast sample with separated lanes.

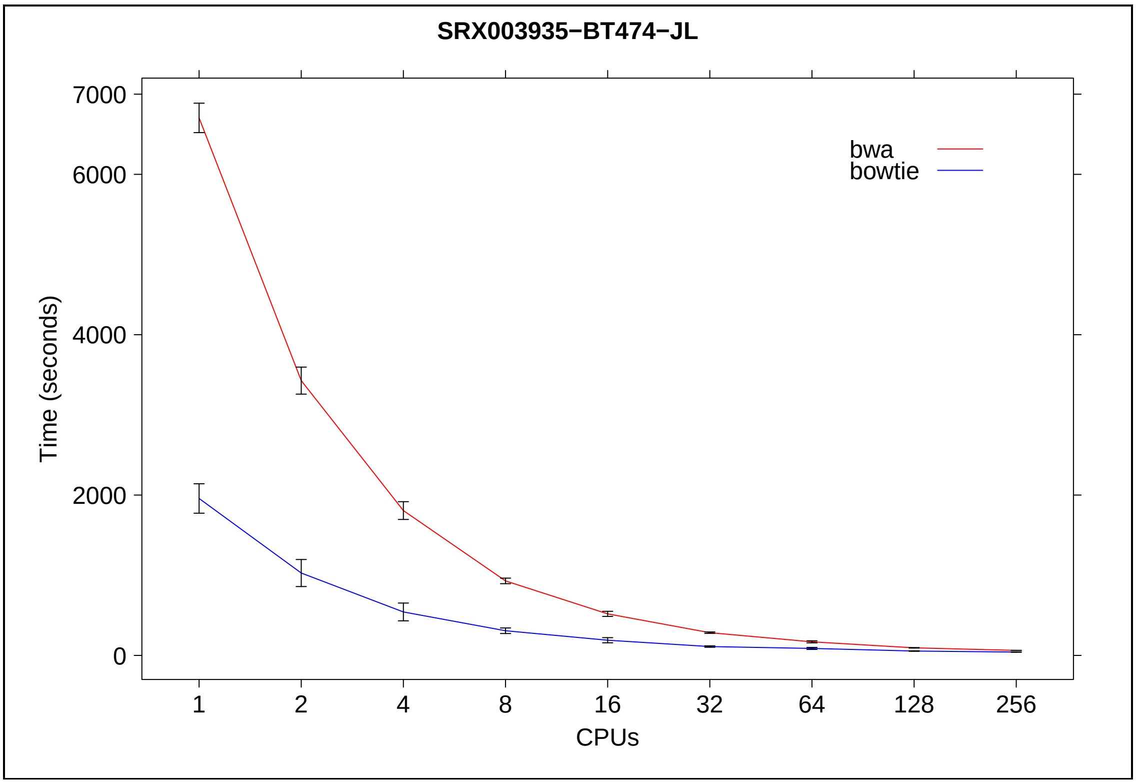


Figure 4.3: Scalability comparison of Bowtie and BWA on merged lanes of BT474 sample.

	BWA		BOWTIE	
CPU	mean	sd	mean	sd
1	6704.00	183.54	1957.50	47.87
2	3427.50	168.54	1027.88	44.76
4	1806.69	110.85	542.44	20.71
8	929.53	34.60	307.88	6.85
16	518.50	32.06	189.70	14.65
32	283.84	8.86	110.62	4.76
64	169.73	12.62	87.32	18.18
128	94.72	2.95	54.90	4.26
256	62.59	1.11	41.72	5.22

Table 4.3: Scalability (in seconds) on BT474 sample with merged lanes.

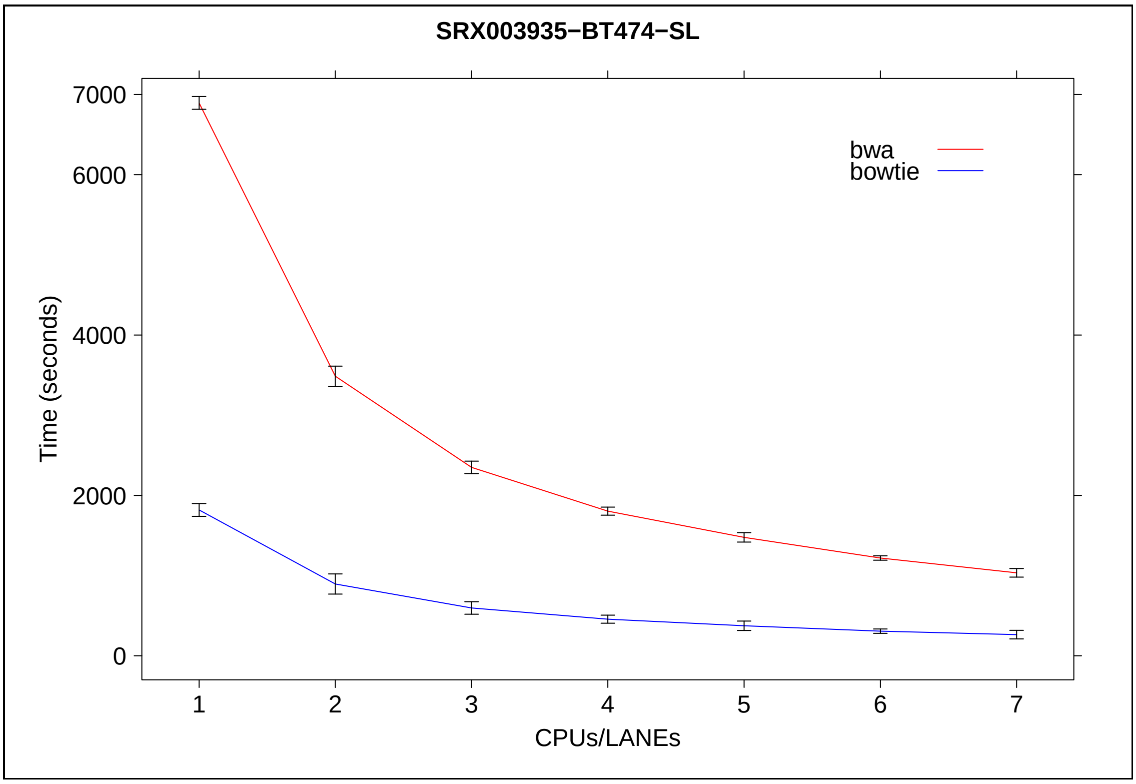


Figure 4.4: Scalability comparison of Bowtie and BWA on separated lanes of BT474 sample.

	BWA		BOWTIE	
lane	mean	sd	mean	sd
1	6895.25	79.73	1819.00	80.91
2	3486.62	125.62	895.12	22.25
3	2349.67	77.84	596.17	5.94
4	1803.88	50.31	456.31	16.88
5	1476.75	58.70	374.25	15.88
6	1219.12	27.12	307.00	16.67
7	1034.61	53.56	263.68	11.06

Table 4.4: Scalability (in seconds) on BT474 sample with separated lanes.

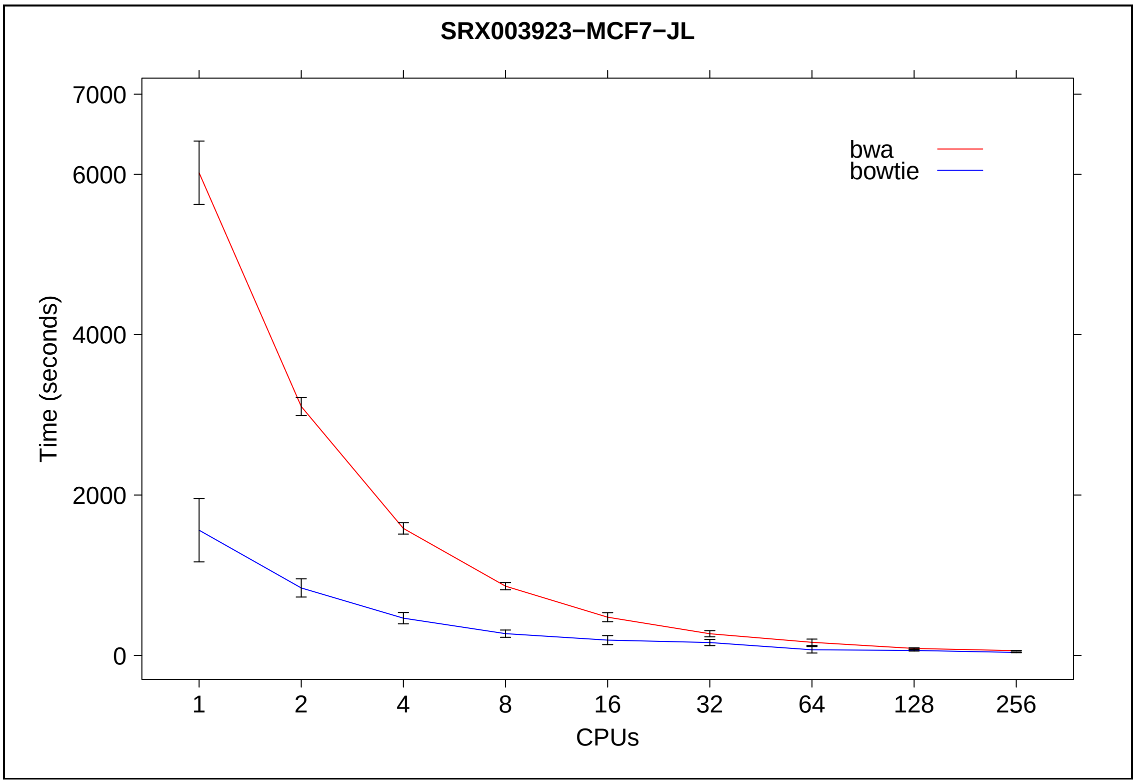


Figure 4.5: Scalability comparison of Bowtie and BWA on merged lanes of MCF7 sample.

	BWA		BOWTIE	
CPU	mean	sd	mean	sd
1	6020.00	395.58	1561.75	92.88
2	3104.75	113.16	841.38	78.12
4	1583.81	70.62	464.56	69.81
8	863.59	45.14	271.50	47.87
16	476.59	56.13	190.89	48.52
32	270.40	38.32	160.72	45.41
64	163.40	40.38	70.22	20.50
128	87.01	7.80	62.08	21.64
256	59.08	3.12	37.17	2.57

Table 4.5: Scalability (in seconds) on MCF7 sample with merged lanes.

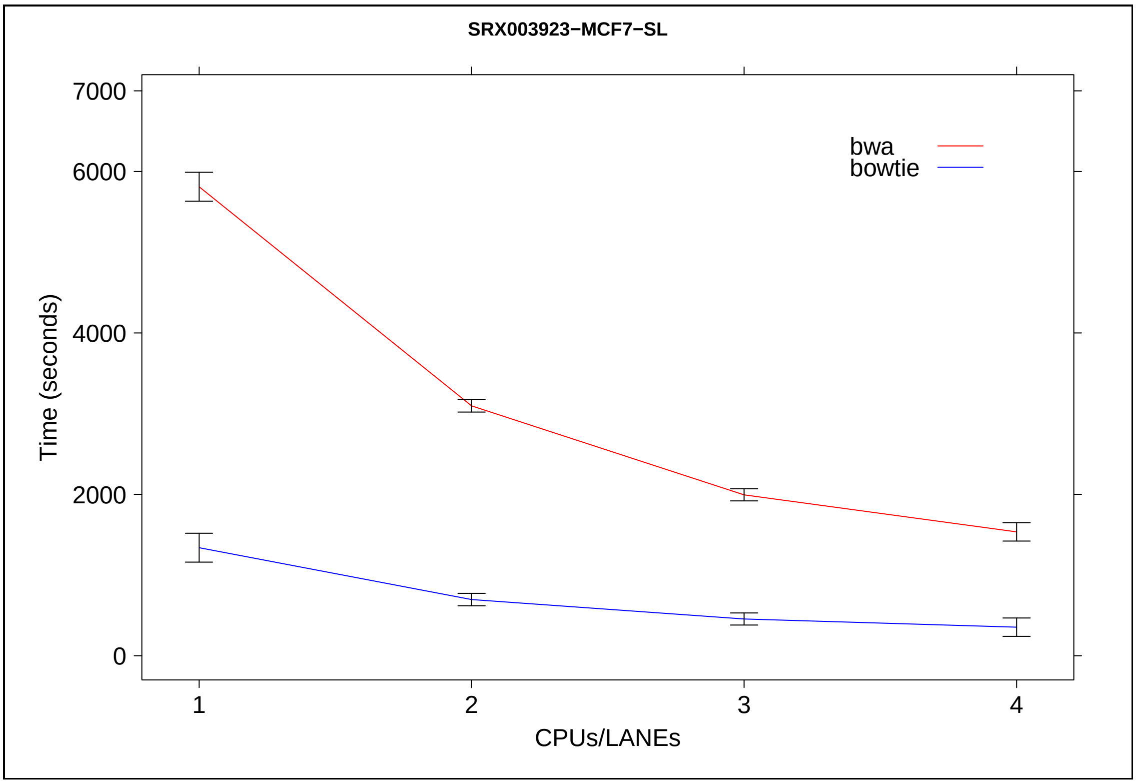


Figure 4.6: Scalability comparison of Bowtie and BWA on separated lanes of MCF7 sample.

	BWA		BOWTIE	
lane	mean	sd	mean	sd
1	5812.00	178.94	1338.75	41.70
2	3096.38	76.93	696.00	18.71
3	1993.42	74.81	455.08	15.31
4	1534.75	113.87	353.69	13.33

Table 4.6: Scalability (in seconds) on MCF7 sample with separated lanes.

	BWA		BOWTIE	
CPU	mean	sd	mean	sd
1	2.065	0.007	1.917	0.014
2	2.054	0.006	1.875	0.028
4	2.025	0.002	1.808	0.027
8	1.982	0.010	1.688	0.015
16	1.899	0.004	1.478	0.022
32	1.755	0.008	1.203	0.014
64	1.523	0.027	0.891	0.010
128	1.212	0.026	0.601	0.017
256	0.874	0.036	0.387	0.011

Table 4.7: Memory usage (in GB) on BT474 sample with merged lanes.

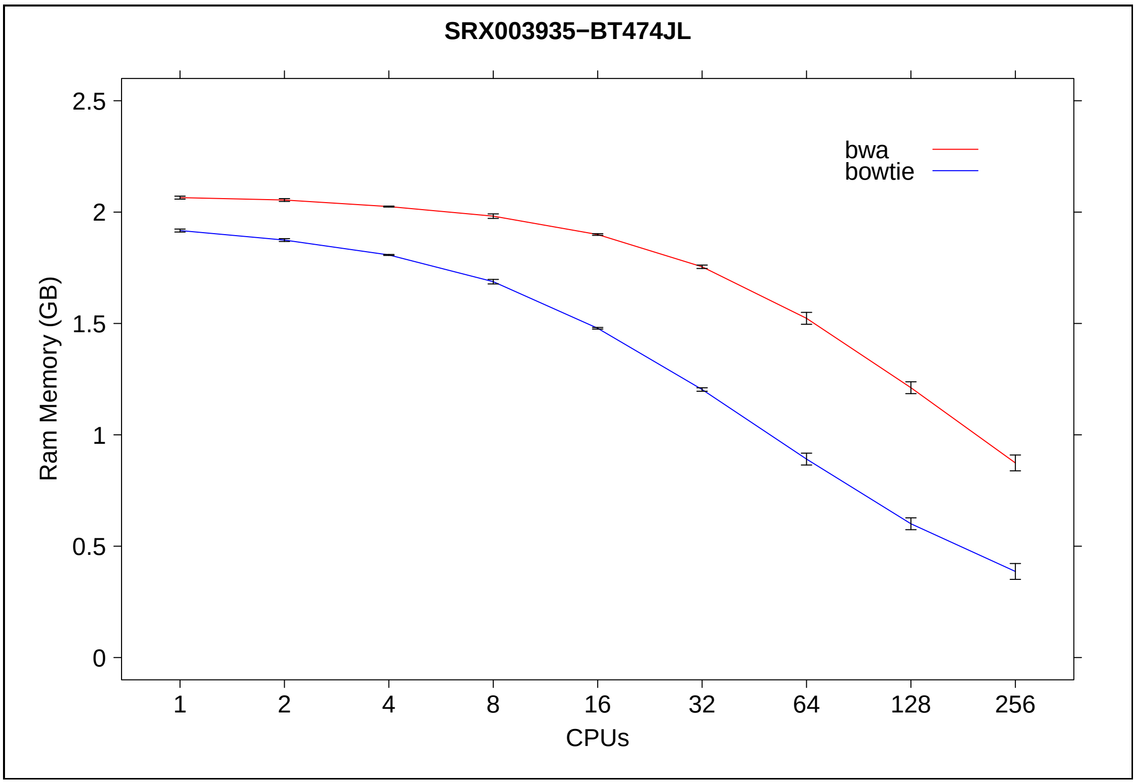


Figure 4.7: Memory usage comparison of Bowtie and BWA on merged lanes of BT474 sample.

	BWA		BOWTIE	
lane	mean	sd	mean	sd
1	2.156	0.011	2.245	0.001
2	2.156	0.010	2.245	0.001
3	2.151	0.004	2.245	0.001
4	2.150	0.003	2.244	0.001
5	2.149	0.005	2.243	0.002
6	2.149	0.003	2.243	0.001
7	2.149	0.002	2.244	0.001

Table 4.8: Memory usage (in GB) on BT474 sample with separated lanes.

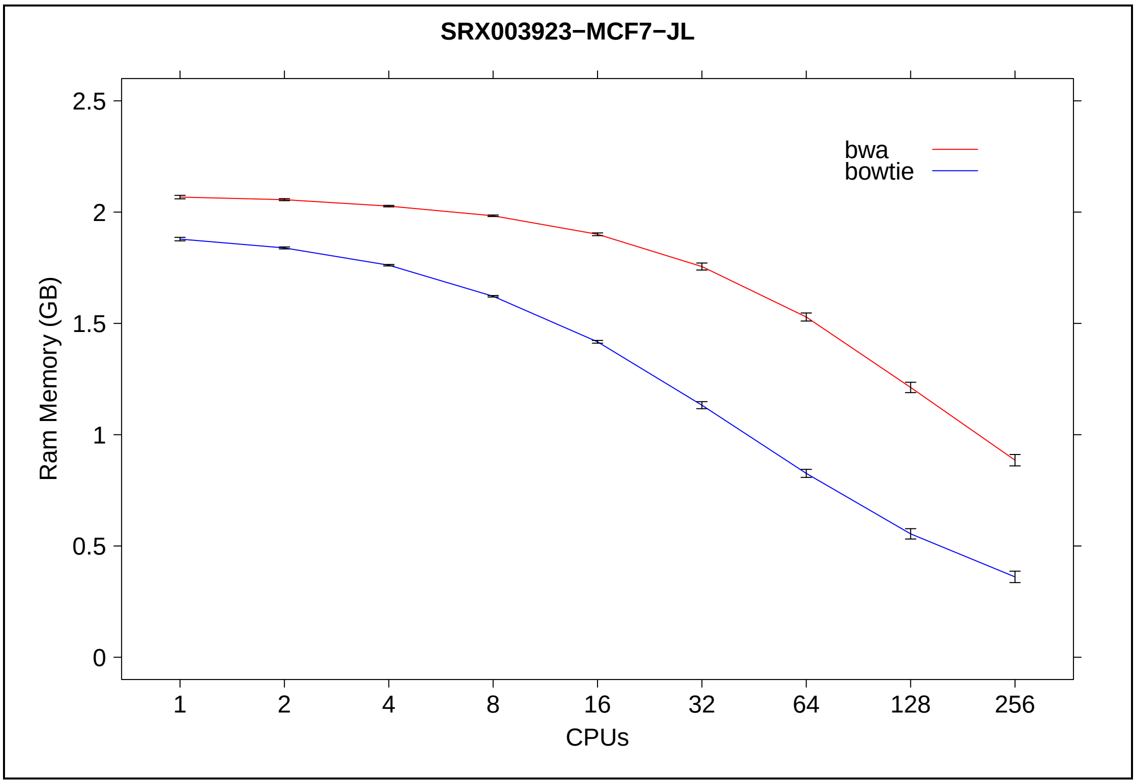


Figure 4.8: Memory usage comparison of Bowtie and BWA on merged lanes of MCF7 sample.

	BWA		BOWTIE	
CPU	mean	sd	mean	sd
1	2.067	0.008	1.879	0.013
2	2.056	0.004	1.839	0.013
4	2.027	0.003	1.762	0.014
8	1.983	0.003	1.622	0.011
16	1.900	0.006	1.417	0.011
32	1.756	0.016	1.133	0.028
64	1.529	0.018	0.826	0.023
128	1.212	0.023	0.555	0.018
256	0.885	0.026	0.361	0.015

Table 4.9: Memory usage (in GB) on MCF7 sample with merged lanes.

	BWA		BOWTIE	
lane	mean	sd	mean	sd
1	2.156	0.002	2.242	0.001
2	2.157	0.008	2.241	0.002
3	2.154	0.004	2.242	0.001
4	2.151	0.003	2.243	0.001

Table 4.10: Memory usage (in GB) on MCF7 sample with separated lanes.

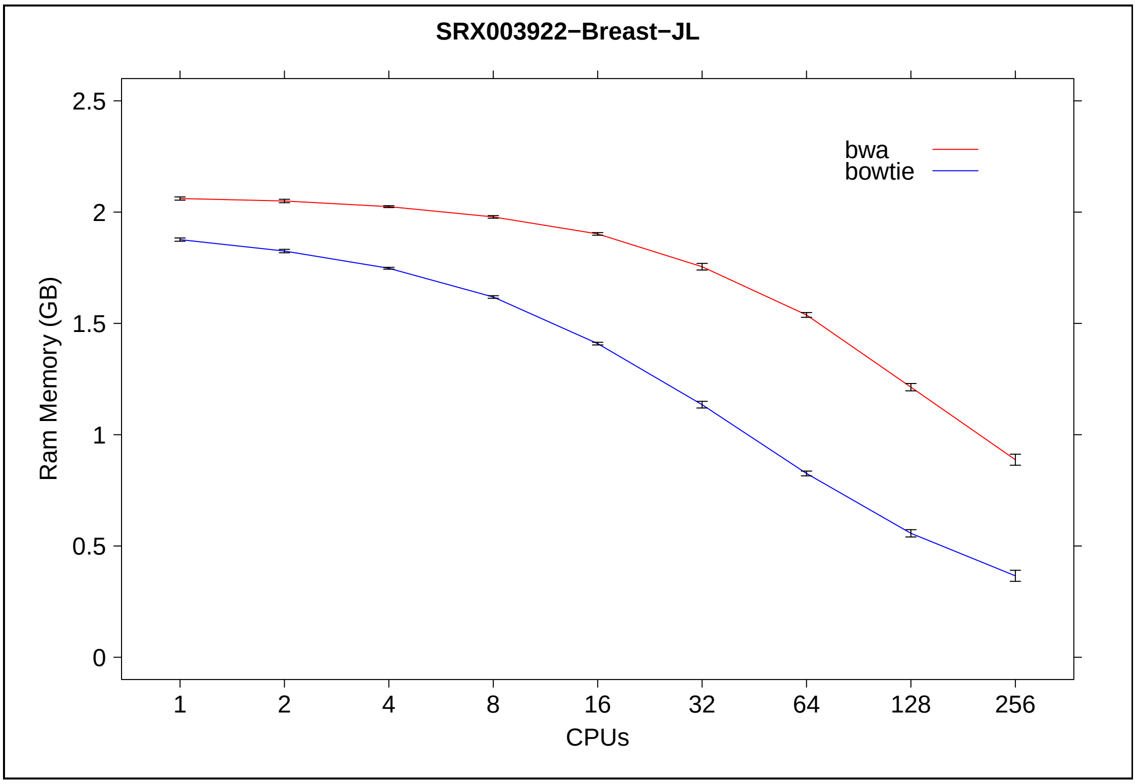


Figure 4.9: Memory usage comparison of Bowtie and BWA on merged lanes of Breast sample.

	BWA		BOWTIE	
CPU	mean	sd	mean	sd
1	2.061	0.007	1.876	0.008
2	2.050	0.008	1.825	0.010
4	2.025	0.004	1.748	0.020
8	1.978	0.006	1.619	0.011
16	1.902	0.006	1.409	0.016
32	1.755	0.015	1.135	0.015
64	1.538	0.011	0.826	0.014
128	1.214	0.016	0.557	0.005
256	0.888	0.025	0.366	0.014

Table 4.11: Memory usage (in GB) on Breast sample with merged lanes.

	BWA		BOWTIE	
lane	mean	sd	mean	sd
1	2.154	0.003	2.242	0.001
2	2.153	0.001	2.242	0.000
3	2.154	0.002	2.242	0.001
4	2.152	0.002	2.242	0.001

Table 4.12: Memory usage (in GB) on Breast sample with separated lanes.

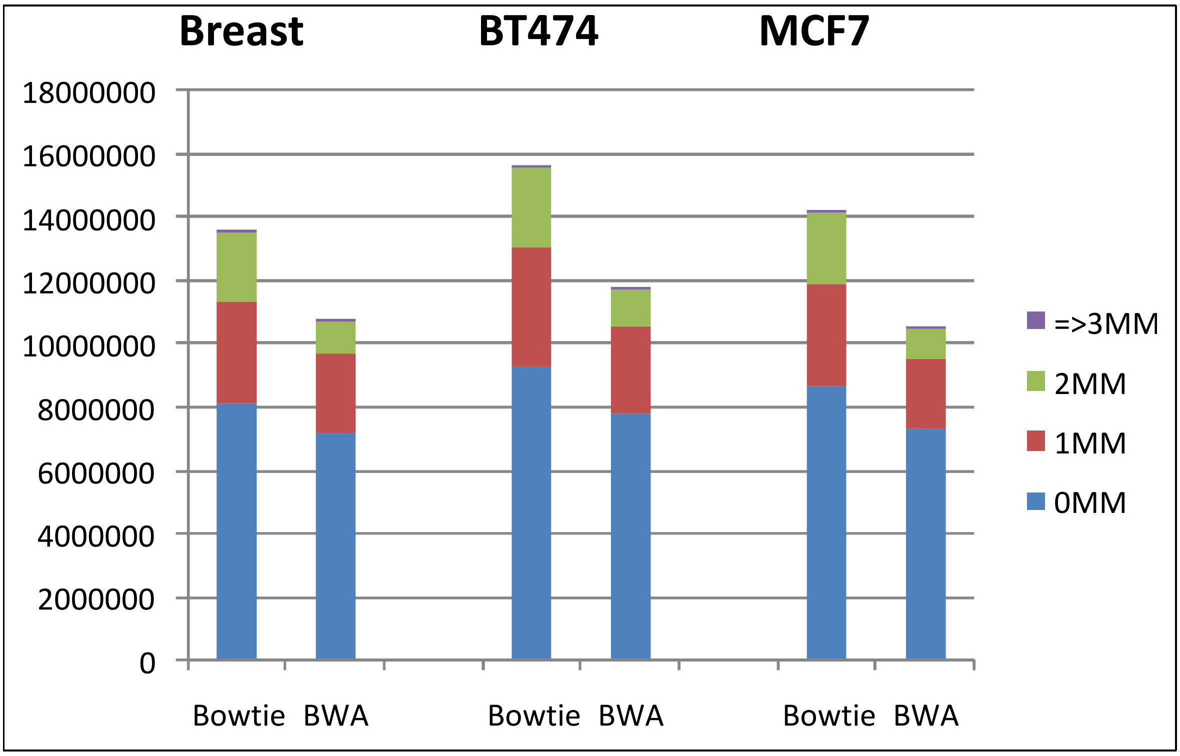


Figure 4.10: Comparison of uniquely mapped reads stratified by the number of allowed mismatches

		total mapped	mismatches + indels			
			0	1	2	3+
BREAST	Bowtie/T	13528932 (83.92%)	8080879	3262783	2171598	13672
	BWA	10730122 (66.56%)	7181244	2528362	1018399	2117
BT474	Bowtie/T	15627171 (84.82%)	9325662	3717151	2532107	52251
	BWA	11694386 (63.47%)	7834663	2724955	1132370	2398
MCF7	Bowtie/T	14161929 (88.18%)	8696849	3171379	2282772	10929
	BWA	10434523 (64.97%)	7320164	2211479	900663	2217

Table 4.13: Uniquely mapped reads and allowed mismatches. The total mapped reads expressed as percentage of the overall number of raw reads in each dataset.

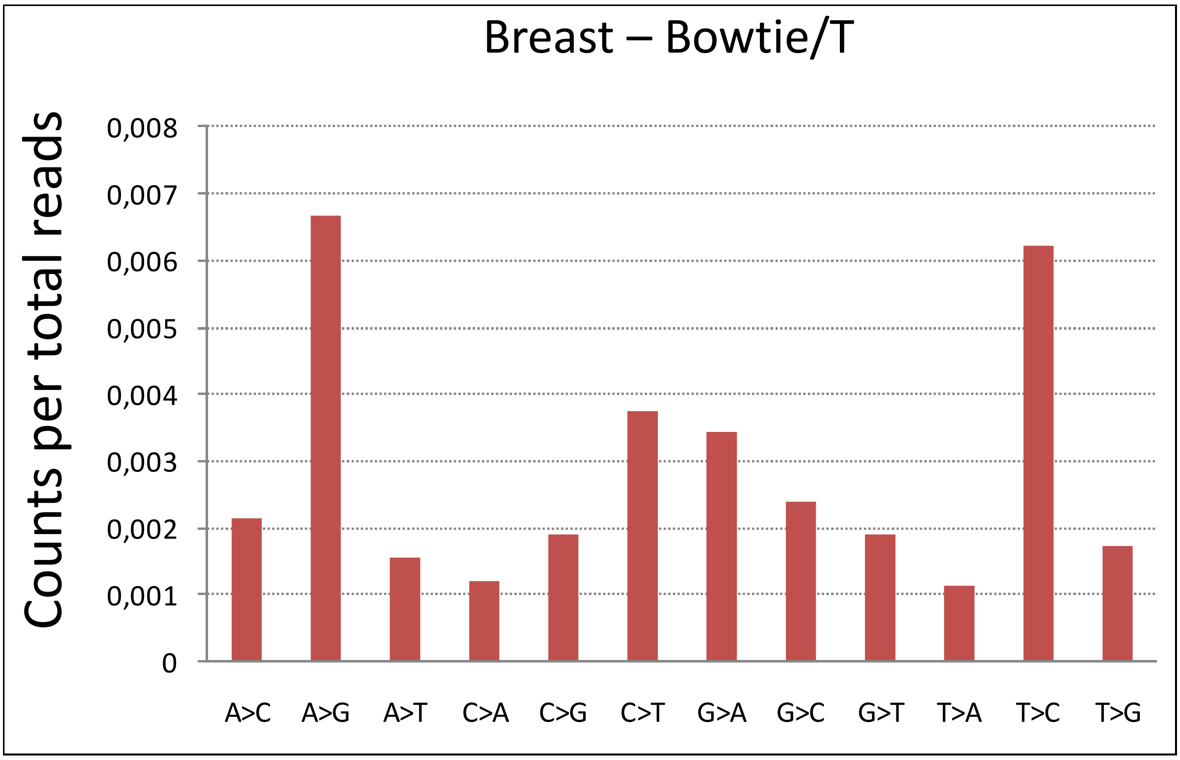


Figure 4.11: Distribution of single base mismatches (merged lanes).

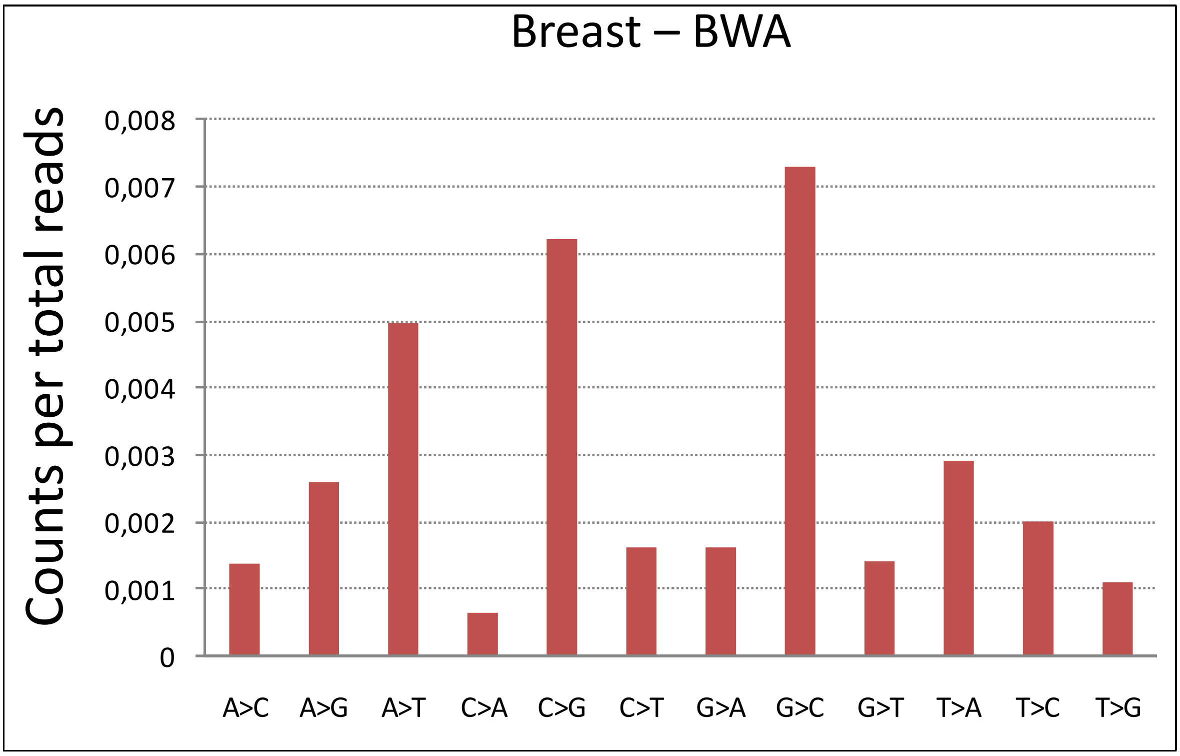


Figure 4.12: Distribution of single base mismatches (merged lanes).

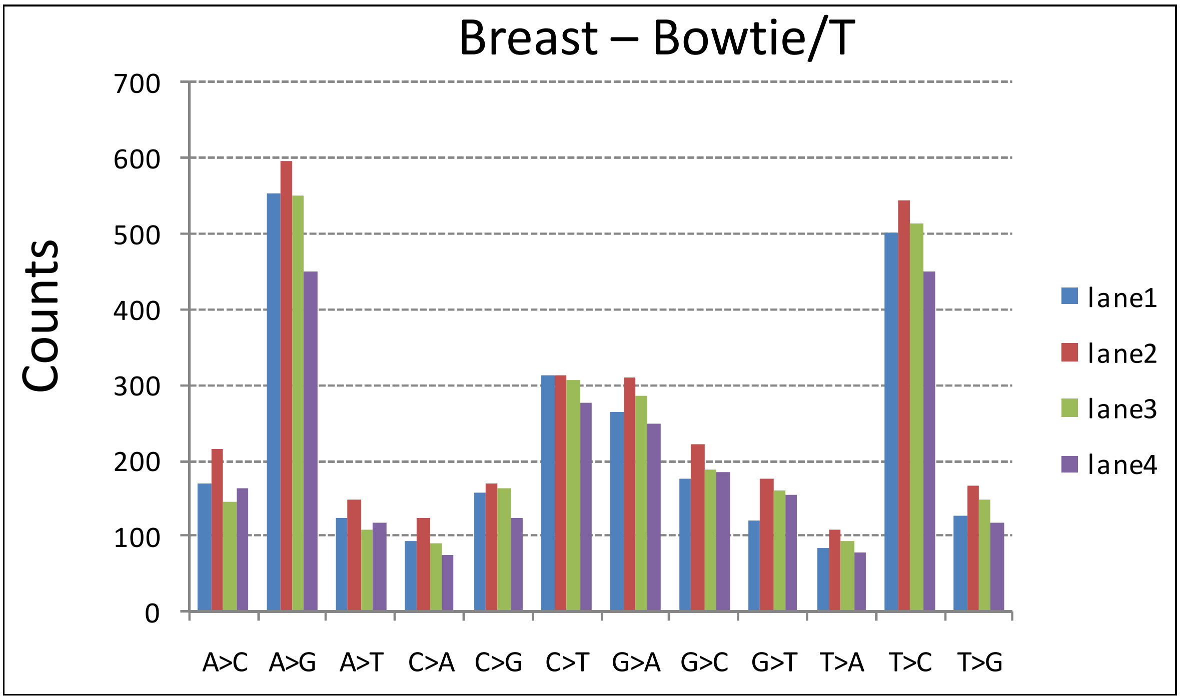


Figure 4.13: Distribution of single base mismatches (separated lanes).

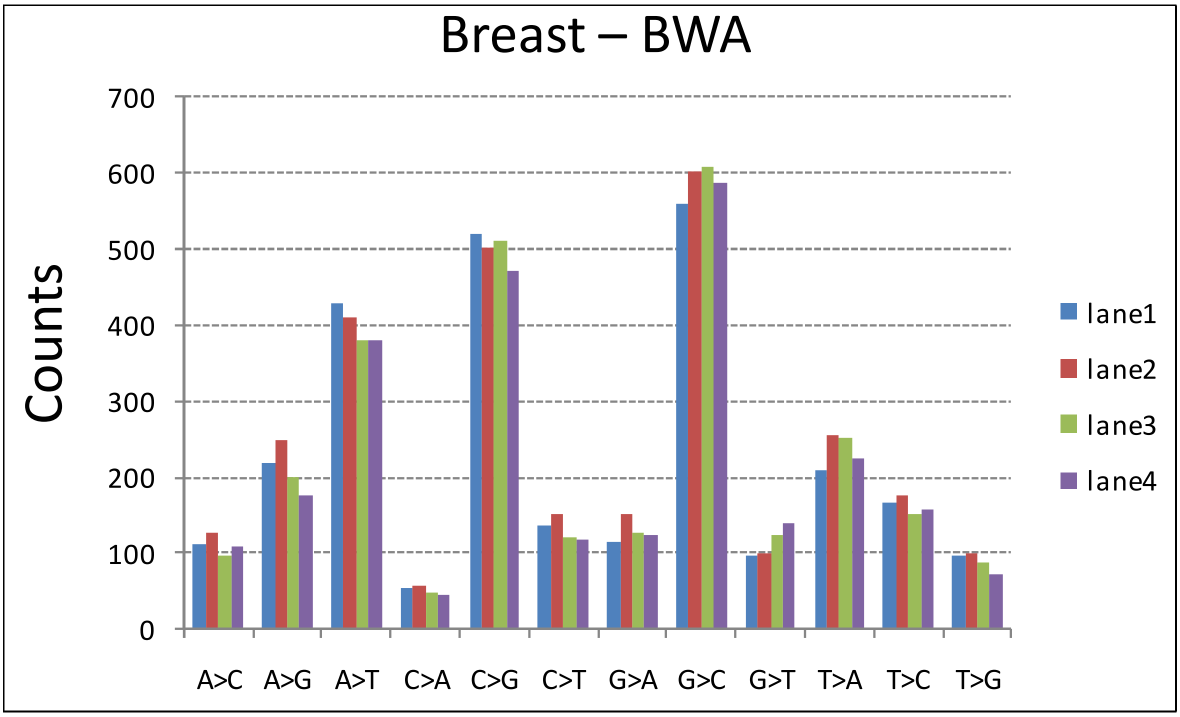


Figure 4.14: Distribution of single base mismatches (separated lanes).

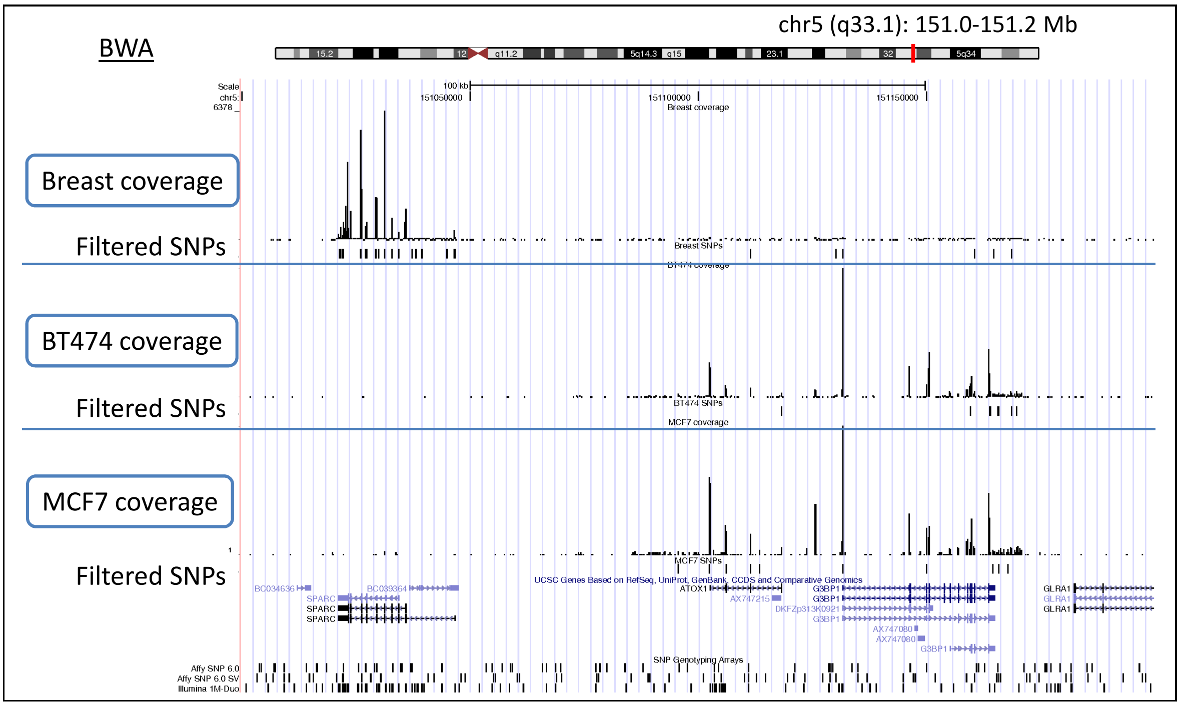


Figure 4.15: Coverage of alignment with Bwa on SPARC, ATOX1 and G3BP1 genes.

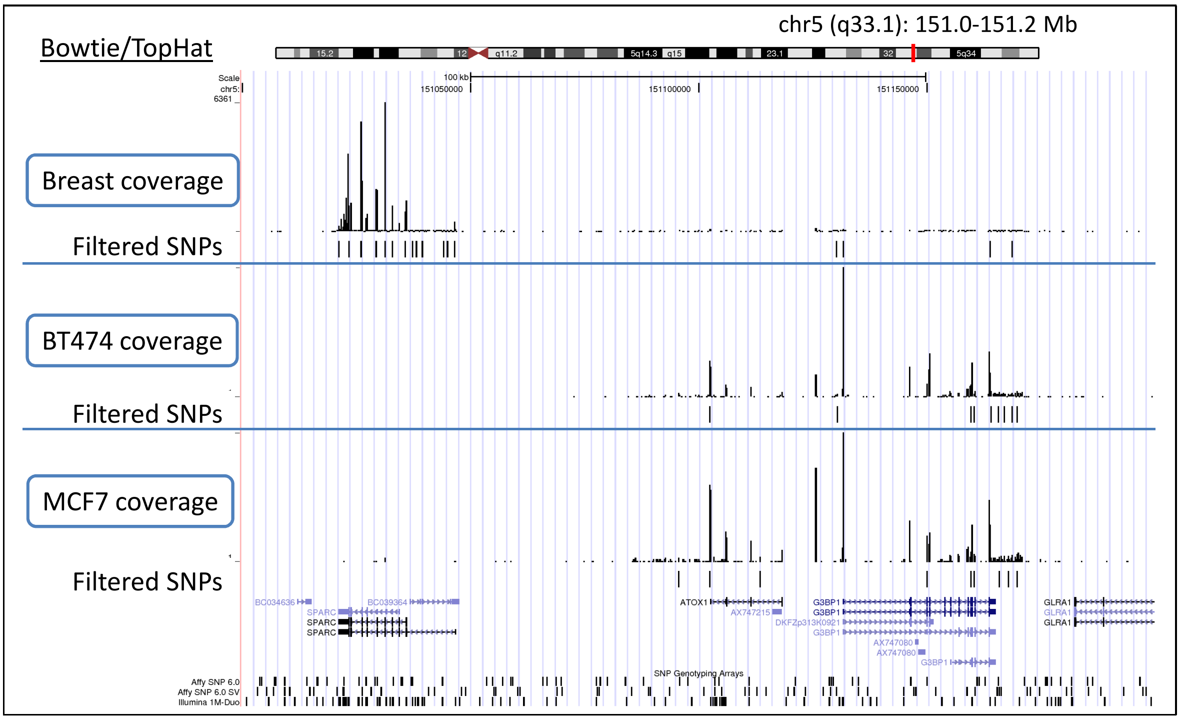


Figure 4.16: Coverage of alignment with Bowtie/Tophat on SPARC, ATOX1 and G3BP1 genes.

4.2 Candidate selection

The first analysis regards the quality the joined output. In the parallelization process the task was split in several sub processes, each one with its input and output, but all outputs have to be joint together eventually. The first result is that the two wrappers, used to parallelize Bowtie and Bwa, return joint outputs that are indistinguishable from the outputs originate by both aligners in "serial mode" (one CPU).

Two methods have been applied in this test suit to parallelize Bowtie and Bwa on the cluster facility. The separated and merged lanes wrappers allow to split and align short reads on several nodes simultaneously. They are quite different: one uses the natural characteristic of the Illumina format and submits one lane to one node directly. Contrariwise, the second counts the sum of all reads from all lanes and divides them in several groups as the number of used CPUs. While the separated lanes method is easier to implement than other, it can use a number of CPUs/nodes equal to the number of lanes, so a limit number of processors can be used: the maximum is 7. On the other hand, the merged lanes procedure allows to use a large number of CPUs and a *high grade of parallelism* can be archived. As shown in Figure 4.1, Figure 4.3 and Figure 4.5 and Figure 4.2, Figure 4.4 and Figure 4.6 using a large number of processors the alignment time decreases. We conclude that in cases where a great number of short reads are included in lane files the merged lanes method is more suitable to speed up investigations.

The RAM memory usage analysis shows that the merged method allocate less memory than the separated one. While in separated approach the RAM consumption is constant, in the other method the RAM utilization decreases with the number of processors. Especially in Bowtie case, this difference is remarkable. This memory reduction can help to limit overload in allocated nodes involved in alignment task.

In time and memory consuming experiments an advantage of Bowtie with respect to Bwa was found on these variances. The differences in computational time is manifest already with low number of CPUs in both wrapper methods. Also in RAM memory probes this distinction exists and Bowtie obtains best values than Bwa.

In quality analysis the comparison of the performance in Figure 4.10 and 4.13 show that Bowtie archives a bigger percentage of uniquely mapped reads than Bwa.

In Figure 4.15 and Figure 4.16 the coverage of alignment from the two aligners are presented. Similar profiles are found for Bwa and Bowtie for genes of significant oncological interest. The SPARC gene is expressed in normal breast and not in the BT474 & MCF7 breast cancer cell lines. Moderate expression of genes ATOX1 and G3BP1 is found in BT474 & MCF7 vs Breast. Neither of the latter two were previously associated with breast cancer susceptibility.

Given its performance in computational time, RAM memory usage and quality in SNP analysis, Bowtie was chosen as the candidate in the D-Daemons architecture.

Chapter 5

Pipelines for reproducibility

The pipeline structure proposed in this thesis for NGS has been motivated by the need of controlling and improving reproducibility of analysis on high throughput data. What follows are two examples where two analysis pipelines have been developed to analyze proteomics data. These works present two implementations focused to produce unbiased and reproducible results.

5.1 A grid-enabled example

In [31] a grid-enabled pipeline with an ontology based environment for proteomics spectra management and a machine learning platform for unbiased predictive analysis is presented. Two existing software platforms (*MS-Analyzer* and *BioDCV*), the emerging proteomics standards, and the middleware and computing resources of the EGEE Biomed VO grid infrastructure[1] are exploited. In the setup, BioDCV is accessed by the MSAnalyzer workflow as a web service, thus providing a complete grid environment for proteomics data analysis.

The environment described in Figure 5.1 is structured in two systems

[1] A geographically distributed network of computational and storage resources connected through the Internet.

connected by a web service: an upstream one (MS-Analyzer), responsible for managing and preprocessing the raw data produced by the spectrometer, and a downstream one (BioDCV), responsible for performing classification and feature ranking inside a complete validation methodology. Web services, workflows, and grid middleware are used to build the infrastructure.

Internet web services are used to remotely integrate the main components of the proposed environment. A web service is a software system designed to support interoperable machine-to-machine interaction over a network. This definition encompasses many different specifications; the standard one is based on SOAP (Simple Object Access Protocol), using messages formatted in XML and sent over the HTTP protocol.

MS-Analyzer uses a Service Oriented Architecture (SOA) and provides a collection of specialized spectra management services, including spectra preprocessing, spectra analysis (obtained by wrapping public available data mining and visualization software tools), and data movement services. The adoption of the SOA approach permits integration into the MS-Analyzer of additional spectra management services (e.g. novel preprocessing tools) and sophisticated, third party analysis tools such as the BioDCV service.

The predictive modeling portion of the proposed system is provided by BioDCV, a platform for machine learning in high-throughput functional genomics. BioDCV fully supports complete validation [39] in order to control selection bias effects, i.e. the generation of optimistic and not reproducible results. For proteomics, it includes methods for baseline subtraction, spectra alignment, peak clustering and peak assignment that were adapted from existing R packages and concatenated to the complete validation system. Since March 2006, BioDCV has been running as an external application in the Egee Biomed VO, the virtual organization for the biomedical domain of the EGEE project [40].

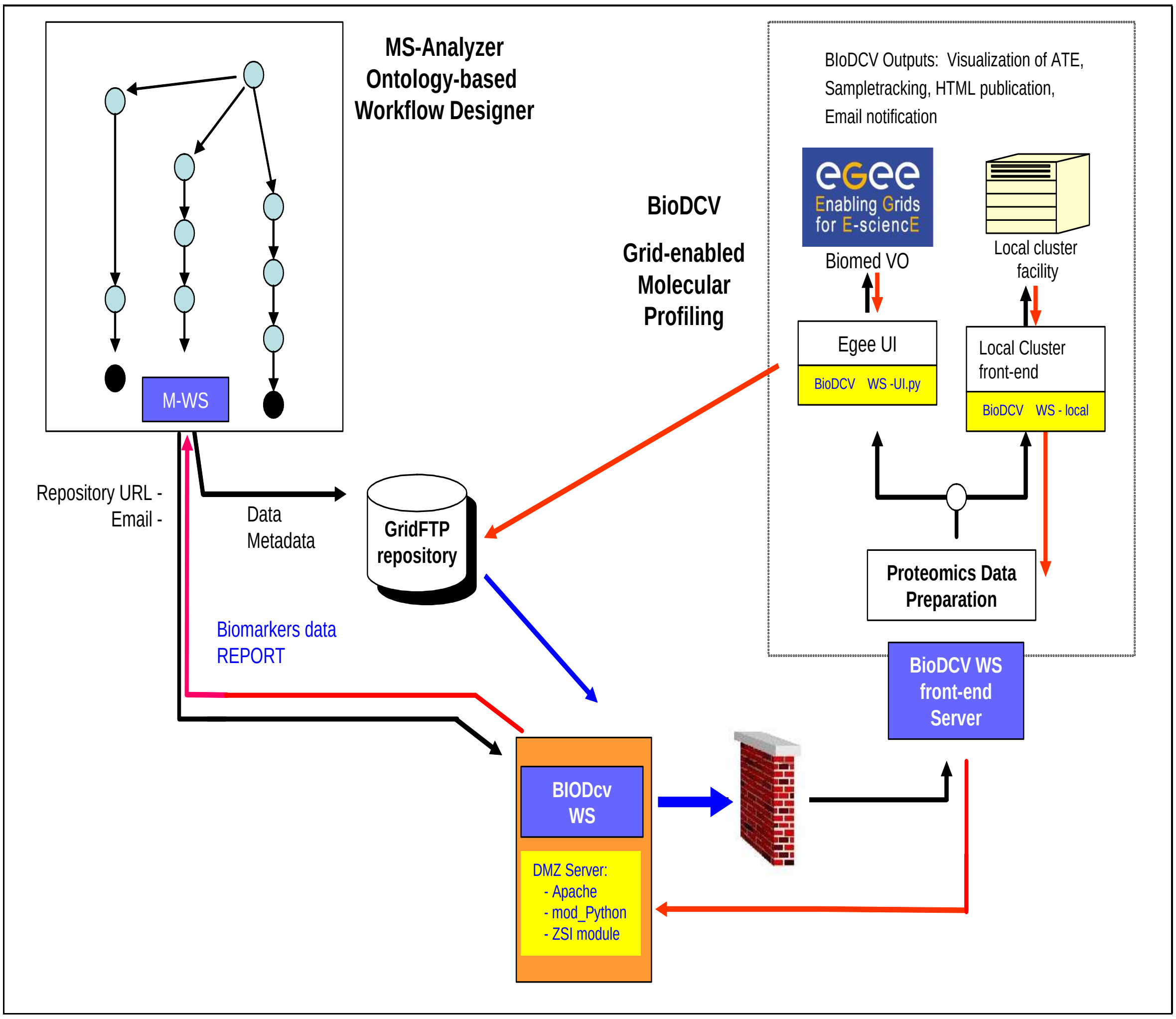


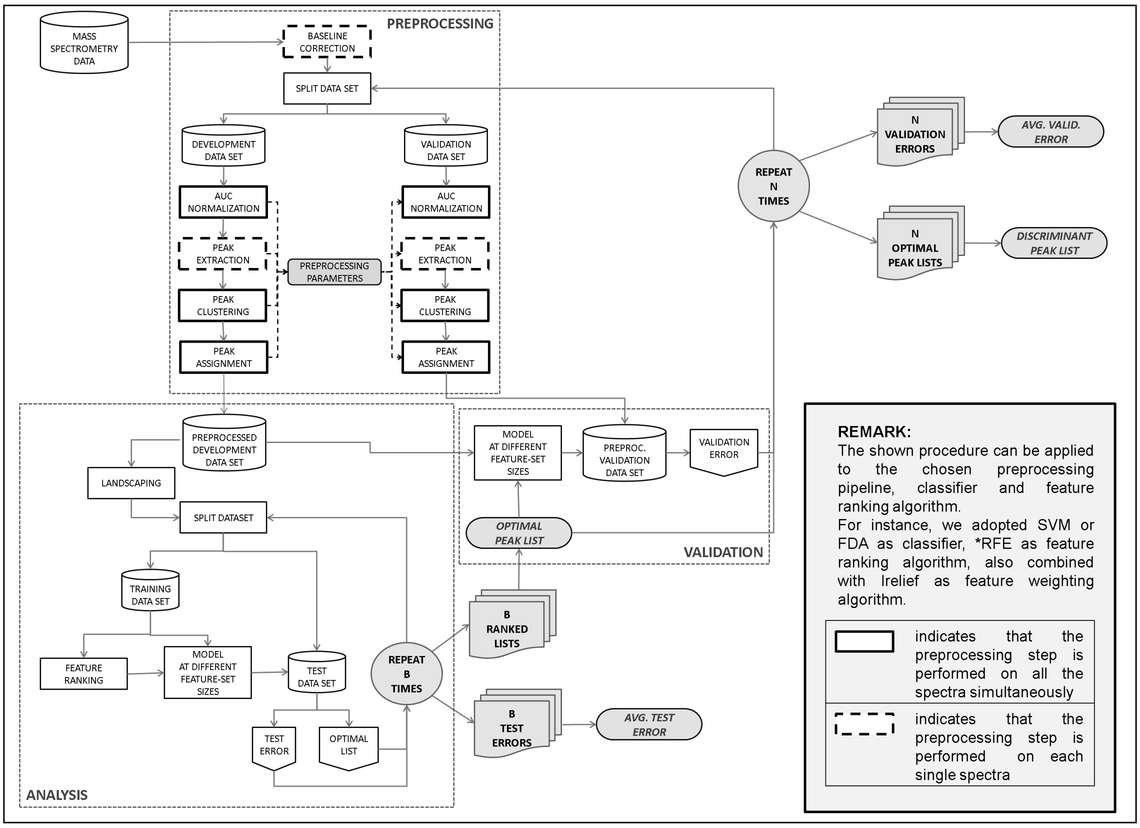
Figure 5.1: The architecture of the MSAnalyzer and BioDCV tools.

5.2 A Design Analysis Protocol

In [41] is described a method to identify predictive biomarkers of disease from high-throughput mass spectrometry (MS) data, which requires a complex analysis path. Preprocessing and machine-learning modules are pipelined, starting from raw spectra, to set up a predictive classifier based on a shortlist of candidate features. As a machine-learning problem, proteomic profiling on MS data needs caution like the microarray case. The risk of overfitting and of selection bias effects is pervasive: not only potential features easily outnumber samples by 103 times, but it is easy to neglect information leakage effects during preprocessing from spectra to peaks. The aim of this review is to explain how to build a general purpose *design analysis protocol (DAP)* for predictive proteomic profiling. The DAP can be used with alternative components, i.e. with different preprocessing methods (peak clustering or wavelet based), classifiers e.g. Support Vector Machine (SVM) or feature ranking methods recursive feature elimination (RFE) [39] or I-Relief. A procedure for assessing stability and predictive value of the resulting biomarkers list is also provided.

The structure of a DAP pipeline for proteomic profiling is displayed in Figure 5.2. The MS spectra are first passed to the preprocessing engine. As soon as possible, we set apart a portion of the data for validation, and then apply a pipeline of preprocessing modules to the development data. Ideally, each step should be applied to spectra separately (dashed boxes in Figure 5.2), i.e. preprocessing parameters should be used unmodified during upstream analysis of validation (e.g. with AUC normalization).

The risk of information leakage is potentially high, thus the DAP implements robust preprocessing methods. Therefore, the solution is to move the preprocessing phase inside the pipeline directly.



REMARK:
The shown procedure can be applied to the chosen preprocessing pipeline, classifier and feature ranking algorithm.
For instance, we adopted SVM or FDA as classifier, *RFE as feature ranking algorithm, also combined with Irelief as feature weighting algorithm.

▭	indicates that the preprocessing step is performed on all the spectra simultaneously
⬚	indicates that the preprocessing step is performed on each single spectra

Figure 5.2: A workflow for proteomic profiling.

Chapter 6

Comparison with existing frameworks

In this section a review of environments for high throughput analysis in genome research is presented to highlight differences and similarities with the D-Daemons solution proposed in Chapter 3.

6.1 miRMaid

MicroRNAs (miRNAs) are short regulatory RNA molecules that are encoded in the genomes of animals, plants and viruses. The latest results in the main mechanism of miRNA based regulation have led to a large set of algorithms, websites and databases that provide different materials about this biology data.

The miRBase website has become a central and highly useful website for scientists that search information about specific miRNAs. Here, scientists can also submit newly discovered miRNAs and details about sequences and homologies in other species. The site includes the following data resources:

1. genomic contexts and evolutionary conservation of miRNAs;

2. prediction and validation of miRNA targets;

3. biological functions and phenotypes of individual miRNAs;

All these miRNA functions are primarily available online as point-and-click websites.

A typical analysis on miRNA (which uses an online repository like miRBase) requires manually downloading raw data files (if available), understanding the format and structure of the resource in question and finally, writing of a script to parse the content and identifiers. A researcher has to go through all these steps, and repeat them each time a resource is updated. *miRMaid* [42] is a software framework designed to simplify the previous manual steps, reduce errors, increase reproducibility of the scientific results and make the data analysis less labor-intensive. It is built in Ruby on Rails (RoR) [43], that allows rapid development of web applications. In the RoR framework, data is stored in a relational database management system (SQLite, PostgreSQL and MySQL are currently supported in miRMaid). miRBase is the data source of the core miRMaid architecture.

miRMaid system uses a RoR layer to expose miRBase data and functions on the web as read-only "RESTful" resources. The RESTful protocol exposes a function or a document with a specified URL, through a HTML request to this URL it is possible to call or to retire the linked resource. This structure allows to retire and call miRBase's experiments or functions making a simple HTML call to a specific url. Therefore, each data model (i.e. Precursor) in miRMaid has resource URLs for listing all objects (/precursors) or a single object (/precursors/hsa-mir-21). Querying a RESTful web service requires that a program is able to generate a HTTP request to the URL that specifies the resource and then parse the response document. Several programming languages and command-line interfaces support these functions. miRMaid can generate HTML and XML documents for all resource URLs and FASTA files for data queries.

The miRMaid solution presents some similarities with the D-Daemons architecture. A public repository is used as data source: D-Daemons retries experiments from the Sequence Read Archive, while miRMaid from the miRBase site. The two solutions try to automatize and simplify long and tedious manual steps required to perform an analysis procedure. Instead, miRMaid does not implement an external analysis pipeline, because it uses miRBase analysis procedures, and has not a GUI to call exposed functions from miRBase site like in the D-Daemons architecture.

6.2 Metagenomic pipeline

In [44] a set of tools is presented to perform a metagenomic analysis. Metagenomics is the study of genetic material recovered from environmental samples directly. A model investigation is composed by a set of steps:

1. Sequence the target sample;
2. Perform a quality control on generated reads;
3. Generate alignments;
4. Conduct a full taxonomic representation analysis (classification).

Software procedures have been developed to automize these phases in the Galaxy framework [22]. Six tools have been organized in a Galaxy section called “Metagenomic analyses” and are listed in Table 6.1.

This solution performs a metagenomic analysis in a web browser directly. A user can upload his sequencing data in a Galaxy site and call each tool to investigate its data. Moreover, the metagenomic procedures can be concatenated in Galaxy workflow to build a pipeline which automatizes all steps.

Fetch taxonomic representation	Fetches taxonomic information from NCBI reference databases
Summarize taxonomy	This utility computes a summary of all taxonomic ranks.
Draw phylogeny	This tool produces a graphical representation of phylogenetic tree in PDF format.
Find diagnostic hits	Produces lists of sequence reads to a particular set of ranks
Find lowest diagnostic rank	Identifies the lowest taxonomic rank for which a metagenomic sequencing read is diagnostic.
Poisson two-sample text	Tests if the number of reads between two taxa is significantly different.

Table 6.1: Tools of the "Metagenomic analyses".

The used reference databases are currently limited to NT and WGS nucleotide sequence collections from NCBI.

The use of the Galaxy framework is similar in Metagenomic pipeline and D-Daemons architecture and it provides a user friendly interface to use procedures. In both cases, advanced workflows based on Galaxy tools can be built to perform investigations. The Metagenomic solution uses the NCBI public repository to retry taxonomic information. No HPC environments and databases are explicitly involved to manage experiments like in D-Daemons architecture.

6.3 mGene.web

The *mGene.web* [45] system is a website which allows to perform the genome-wide prediction of protein coding genes from eukaryotic DNA sequences. It provides a web interface to the *mGene* gene finding software [45]. mGene is based on discriminative machine learning techniques and its high accuracy has been demonstrated in an international competition on nematode genomes [45]. The web server mGene.web includes a convenient interface to mGene for a use within the Galaxy framework ([22]), which also offers handy access to existing genome annotation databases as well as other computational tools.

The mGene.web system offers pretrained models for the recognition of gene structures including untranslated regions in an increasing number of organisms. For organisms in the list of pre-trained models, a FASTA file with the DNA sequence is required as input and a GFF3 file containing gene predictions is produced as output. In case one wishes to annotate a genome for which no suitable pre-trained model exists, mGene.web calls the mGene functions to train a new model. This phase takes a FASTA file with the DNA and a GFF3 file with a set of known genes as input and returns a trained mGene predictor object that can be used to predict genes on given DNA sequences.

The web service uses a cluster with 84 AMD Opteron CPUs (2.2 GHz) with 8GB of RAM per four CPUs. On this HPC facility the mGene training and prediction tasks are split into several parallel sub-tasks in order to reduce the waiting time for users.

The mGene.web offers a computational gene finding system with the follows features:

1. high accuracy;

2. genome-wide predictions within a reasonable time;

3. easy to use even for researchers with no programming experience;

4. applicable to a large variety of newly sequenced organisms.

In a confront between this solution and the D-Daemons architecture some comparable characteristics can be found. The two systems adopt the

Galaxy framework to service a GUI for researchers with no programming experience. The solutions allow to build workflows with their procedures and other existing Galaxy tools. As in D-Daemons answer a HPC facility is used to reduce waiting time in the heavy computational tasks (mGene processes and HPC pipeline parallel jobs). In mGene.web no public repositories or database system are used in its internal architecture.

Chapter 7

Conclusions

7.1 Overview

Public access sites are growing to allow exchanges of next generation sequencing data sets between research laboratories. This is a great goal considering that in general an UHTS experiment weights several gigabytes. These public repositories give a great opportunity to get access to a large numbers of experiments and use them in own research projects (see Section 3.1). The Sequence Read Archive (SRA) is the most notable, because it is supported by three biology centers, the National Center for Biotechnology Information (NCBI), the European Bioinformatics Institute (EBI) and the DNA Dsata Bank of Japan (DDBJ), and it manages about 11 Terabases of biological sequence data [1].

The SRA presents a search engine called *E*ntrez cross-database search, where a user can insert a research key and the system presents relative archived experiments, then user can download them. All these operations are *manually* and a user have to repeat them to search new data or update previous queries. If the research rate increases, the previously manual steps can take time and patience.

This dissertation has presented a system that can *automatically* process

[1] march 2010

the steps requested to download data from the SRA repository. The D-Daemons architecture can download experiments based on user research key and this operation can be scheduled to retire new data. Second, it processes them on a HPC pipeline and publishes RPKM measures, new junction identification for each genes and compatible files to visualize the aligned short reads on the HG18 chromosome positions in Genome Browser site. These results allow to compare experiments from different biological samples to identify variability and similarity in gene expression.

Several issues have been addressed to build the whole system. First, a knowledge of the SRA architecture has been needed to build the automatic procedures. A considerable amount of time has been spent to find information on how the SRA is organized, where experiments are stored and how the Entrez search engine [34] can be used in an automatic retrieve system. All these details have been applied in the development of the D-Daemons architecture. Second, software for UHTS studies uses different file formats for its input and output. It introduces a complication if your purpose is to concatenate these tools in an analysis pipeline. Therefore, to build the HPC pipeline based on Bowtie, SAMtools and Cufflinks (described in Chapter 2) some conversion operations have been applied to make compatible the input and output between tools.

The integration of the D-Daemons architecture in the Galaxy framework has been performed (see Chapter 3). A graphic user interface (GUI) helps to command the D-Daemons procedures easily. This is an example that shows how a visual interface can make useful a complicate system. This is an important point in software architecture developed for usage in a interdisciplinary environment. A GUI does not only help to use a program or system, but also it reduce training time to acquire knowledge on a instrument without the requisite to understand how all its parts run. Visual frameworks, as Galaxy, provide features (for example the workflow design and execution) to hide complex architectures and present only the main methods. In bioinformatics it has become a requirement, because it is an example where software instruments can be used by biologists and doctors with different computer science knowledge.

7.2 Innovative features

The D-Daemons architecture introduces some innovative features.

1. It is a service that automatically calls the NCBI Entrez tools, to interrogate and retrieve experiments from the SRA public repository.
2. It is built in two components (web-dae and HPC-dae), which can be installed on a HPC infrastructure, different machines or one workstation.
3. The communication between the two components it is based on a TCP/IP socket layer that is compliant with firewalled local area networks.
4. The D-Daemons architecture does not depend by a particular queue system; therefore it can be installed on HPC environment with different resource manager software.

7.3 Improvements

Even though the proposed system provides a working solution and it is indeed used in a research context, several enhancements can be introduced to improve the efficiency and add new features.

Some elements in the HPC pipeline are not natively designed to work in a parallel environment like a cluster facility. The workaround used in the D-Daemons architecture is to use a software layer called *wrapper* to join the concurrent execution feature to these components. This method allows to run these serial software with a good efficiency. Better performances can be archived using tools developed to run in a HPC system natively. A software written to take advantaged of several execution nodes can obtain a better scalability when the number of machines increases than a wrapper version. This upscaling is possible because *parallel software* can use special libraries to divide and coordinate algorithms running on several CPUs of different nodes. The synchronization of software instances is managed through exchanged messages on the network that joins the execution nodes. One of these libraries is the Message Passing Interface (MPI). It is an application programming interface (API) which implements the communications between software instances distributed on a network of computer nodes. One element that consumes a huge execution time is the alignment phase of the short reads. A decisive improvement could be the availability of a parallel implementation of the Burrows-Wheeler Transform algorithm (used in Bowtie and Bwa) based on the MPI library.

Second, the D-Daemons system is based on a flat file database that makes searching for studies, experiments and sample tissues possible through SQL queries. But if the data increases to several terabytes of experiments, a special storage architectures have to be chosen. Storage area network (SAN) and grid/cloud storage systems are examples of storage system that can manage terabyte size of information. A SAN is a set of storage devices that communicate and share data through a local area network. In a grid/cloud storage system components are storage devices and also SANs geographically distributed that expose the storage space through the Internet network. A simple flat file database does not require to manage data over these large storage architectures. A solution is to substitute the flat file database with a *database management system (DBMS)*, which supports services to operate in SANs and cloud environments natively. A DBMS can list experiments over the storage devices and support fault-tolerant functions to protect and backup data.

A DBMS can be also inserted in the D-Daemons architecture if the system has to service a high number of users simultaneously. In a multi-user environment the HPC-dae (see Chapter 3) is the element which has to manage different requests from multiple web-dae instances. This feature is already supported by the Python socket procedures [46] defining HPC-dae. Contrariwise, the SQLite system originally available with Galaxy manages different user requests locking the database when a user makes a write operation and other users can only execute read operations. A SQLite database is a file and the lock procedure is performed through the operating system's file locking support. This mode penalizes SQLite performances in a high ratio of writes to reads and a heavy user load. As an alternative, a DBMS is developed to support hundreds user transactions without penalizing the performance and response time of whole system.

7.4 Future work

The experience with the Galaxy framework (see Chapter 3) has positively confirmed the stability and validity of visual interfaces and workflows in the application context. Their potential in making easy the interaction with complex system to multidisciplinary users is convincing. Further software applications in bioinformatics is expected to operate natively in these visual interfaces and workflows environments: the D-Daemons architecture could be further developed with user intentions in research.

Second, the software tools used in NGS typically run on workstations and HPC facilities based on standard CPU architecture (Intel or AMD processors). Recently, a new hardware platform to execute heavy computational tasks efficiently is introduced. This is indicated as *general-purpose* *computing on graphics processing units (GPGPU)* and it concerns the use of the graphics processing unit (GPU) on the graphic cards, inside a workstation or in a special enclosure. Modern graphical cards have GPUs with a number of cores larger than in standard CPUs, a GPU has hundred cores while the last CPU have only 4 cores (Quadcore processor) and all GPU's cores hold the space of a graphic card in a single workstation. This increase in computational granularity can be used to accelerate parallel software efficiently provided that enough local memory can be addressed and managed. Moreover, to take advantage from the GPU hardware software applications have to be rewritten with special libraries (as the CUDA [47] or the OpenCL framework [48]). Examples of computational resources based on GPUs can be found in several scientific tasks: high energy physic, molecular dynamic, computational chemistry and others. Extension lists are given with the GPGPU website [49] and, recently, at the "CUDA Bio-Informatics and Life Sciences" dedicated resource [50]. The application of the GPGPU technology to speedup algorithms in next generations sequencing, as a potential short read aligner for human reference genome, should thus be seriously examined in the future.

Bibliography

[1] E. T. Wang, R. Sandberg, S. Luo, I. Khrebtukova, L. Zhang, C. Mayr, S. F. Kingsmore, G. P. Schroth, and C. B. Burge. Alternative isoform regulation in human tissue transcriptomes. *Nature*, 456(7), November 2008.

[2] UCSC Genome Browser. `http://genome.ucsc.edu/index.html` .

[3] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology*, 10(Issue 3), March 2009.

[4] R. Li, Y. Li, K. Kristiansen, and J. Wang. SOAP: short oligonucleotide alignment program. *Bioinformatics*, 24(5), January 2008.

[5] H. Li, J. Ruan, and R. Durbin. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res.*, 18r, August 2008.

[6] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg. Fast and accurate short read alignment with BurrowsWheeler transform. *Bioinformatics*, 25(14), May 2009.

[7] R. Li, C. Yu, Y. Li, T. Lam, S. Yiu, K. Kristiansen, and J. Wang. SOAP2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25(15), May 2009.

[8] C. Trapnell, L. Pachter, and S. L. Salzberg. Tophat: discovering splice junctions with RNA-Seq. *Bioinformatics*, 25(9), March 2009.

[9] A. Mortazavi, B. A. Williams, K. McCue, L. Schaeffer, and B. Wold. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature methods*, 5(7), July 2008.

[10] NCBI Sequence Read Archive. `http://www.ncbi.nlm.nih.gov/Traces/sra` .

[11] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, and 1000 Genome Project Data Processing Subgroup. The sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16), August 2009.

[12] M.R. Green. PRE-mRNA SPLICING. *Annual Review Genetics*, 20, 1986.

[13] BAM Format. `http://genome.ucsc.edu/goldenPath/help/bam.html` .

[14] Cufflinks tool. `http://cufflinks.cbcb.umd.edu` .

[15] H Jiang and W. H. Wong. Statistical Inferences for isoform expression. *Bioinformatics*, 25(8), 2009.

[16] Gene transfer format. `http://mblab.wustl.edu/GTF22.html` .

[17] B. Langmead, M. C. Schatz, J. Lin, M. Pop, and S. L. Salzberg. Searching for SNPs with cloud computing. *Genome Biology*, 10(11), 2009.

[18] R. Li, Y. Li, X. Fang, H. Yang, J. Wang, and K. Kristiansen. Snp detection for massively parallel whole-genome resequencing. *Genome Res*, 19, 2009.

[19] Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2 .

[20] W.J. Kent, C. W. Sugnet, T. S. Furey, K.M. Roskin, T. H. Pringle, A. M. Zahler, and D. Haussler. The human genome browser at UCSC. *Genome Res.*, 12, 2002.

[21] The ENCODE Project Consortium. Identification and analysis of functional elements in 1% of the human genome by the encode pilot project. *Nature*, 447, June 2007.

[22] B. Giardine, C. Riemer, R. C. Hardison, R. Burhans, L. Elnitski, P. Shah, Y. Zhang, D. Blankenberg, I. Albert, J. Taylor, W. Miller, W. J. Kent, and A. Nekrutenko. Galaxy: A platform for interactive large-scale genome analysis. *Genome Res*, 15, 2005.

[23] Python language. http://www.python.org .

[24] SQLite: a self-contained, embeddable, zero-configuration SQL database engime. http://www.sqlite.org .

[25] INSDC: International Nucleotide Sequence Database Collaboration. http://www.insdc.org .

[26] M. Shumway, G. Cochrane, and H. Sugawara. Archiving next generation sequencing data. *Nucleic Acids Research*, 38 (Database issue), 2010.

[27] E.W. Sayers, T. Barrett, D.A. Benson, S.H. Bryant, K. Canese, V. Chetvernin, D.M. Church, M. DiCuccio, R. Edgar, S. Federhen, M. Feolo, L.Y. Geer, W. Helmberg, Y. Kapustin, D. Landsman, D.J. Lipman, T.L. Madden, D.R. Maglott, V.Miller, I. Mizrachi, J. Ostell, K.D. Pruitt, G.D. Schuler, E. Sequeira, S.T. Sherry, M. Shumway, K. Sirotkin, A. Souvorov, G. Starchenko, T.A. Tatusova, L. Wagner, E. Yaschenko, and J. Ye. Database resources of the national center

for biotechnology information. *Nucleic Acids Research*, 37 (Database issue), 2009.

[28] EBI Sequence Read Archive. `http://www.ebi.ac.uk/ena` .

[29] DDBJ Sequence Read Archive. `http://www.ddbj.nig.ac.jp/sub/trace_sra-e.html` .

[30] 1000 Genomes: A Deep Catalog of Human Genetic Variation. `http://www.1000genomes.org` .

[31] M. Cannataro, A. Barla, R. Flor, A. Gallo, G. Jurman, S. Merler, S. Paoli, G. Tradigo, P. Veltri, and C. Furlanello. A grid environment for high-throughput proteomics. *IEEE Transactions on Nanobioscience*, 6(2), 2007.

[32] Biopython: Python tools for computational molecular biology. `http://biopython.org/wiki/Main_Page` .

[33] Machine Learning PY. `https://mlpy.fbk.eu` .

[34] Entrez Programming Utilities. `http://eutils.ncbi.nlm.nih.gov`.

[35] bedGraph Track Format. `http://genome.ucsc.edu/goldenPath/help/bedgraph.html` .

[36] bigWig Track Format. `http://genome.ucsc.edu/goldenPath/help/bigWig.html` .

[37] Sun Grid Engine. `http://gridengine.sunsource.net` .

[38] Fondazione Bruno Kessler. `http://www.fbk.eu` .

[39] C. Furlanello, M. Serafini, S. Merler, and G. Jurman. Semisupervised learning for molecular profiling. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(2), 2005.

[40] Enabling Grids for E-sciencE (EGEE). http://www.eu-egee.org .

[41] A. Barla, G. Jurman, S. Riccadonna, M. Chierici, S. Merler, and C. Furlanello. Machine learning methods for predictive proteomics. *Briefings in Bioinformatics*, 9(2), 2008.

[42] A. Jacobsen, A. Krogh, S. Kauppinen, and M. Lindow. miRMaid: a unified programming interface for microRNA data resources. *BMC bioinformatics*, 11(1), 2010.

[43] Ruby on Rails home. http://www.rubyonrails.org .

[44] K. Pond, S. Wadhawan, F. Chiaromonte, G. Ananda, W.Y. Chung, J. Taylor, and A. Nekrutenko. Windshield splatter analysis with the Galaxy metagenomic pipeline. *Genome Research*, 2009.

[45] G. Schweikert, J. Behr, A. Zien, G. Zeller, C. S. Ong, S. Sonnenburg, and G. Rätsch. mgene.web: a web service for accurate computational gene finding. *Nucleic Acids Research*, 38 (Database issue), 2009.

[46] Python and socket documentation. http://docs.python.org/library/socket.html .

[47] CUDA ZONE. http://www.nvidia.com/object/cuda_home_new.html .

[48] OpenCL Overview. http://www.khronos.org/opencl .

[49] General-Purpose Computation on Graphics Hardware. http://gpgpu.org .

[50] CUDA Bio-Informatics and Life Sciences. http://www.nvidia.com/object/bio_info_life_sciences.html .

Appendix A

Academic and technical papers by the candidate

Journals

1. G. Jurman, S. Merler, A. Barla, **S. Paoli**, A. Galea, and C. Furlanello. Algebraic stability indicators for ranked lists in molecular profiling. *Bioinformatics*, 24(2):258-264, 2008.

2. **S. Paoli**, G. Jurman, D. Albanese, S. Merler, and C. Furlanello. Integrating gene expression profiling and clinical data. *Int. J of Approximate Reasoning*, 47(1):58-69, 2008.

3. S. Riccadonna, G. Jurman, S. Merler, **S. Paoli**, A. Quattrone, and C. Furlanello. Supervised classification of combined copy number and gene expression data. *Journal of Integrative Bioinformatics*, 4(3):74, 2007.

4. M. Cannataro, A. Barla, R. Flor, A. Gallo, G. Jurman, S. Merler, **S. Paoli**, G. Tradigo, P. Veltri, and C. Furlanello. A grid environment for high-throughput proteomics. *IEEE Transactions on Nanobioscience*, 6(2):117-123, 2007.

International conferences with Review Committee

- M. Chierici, M. Roncador, **S. Paoli**, G. Jurman, and C. Furlanello. Comparing uhts pipelines for snp discovery from rna-seq data. *MGED 12*, October 5 - 8, 2009, Phoenix, Arizona, USA

- S. Riccadonna, D. Albanese, **S. Paoli**, G. Jurman, and C. Furlanello. The mlpy/biodcv machine learning environment for reproducible molecular signatures. *MGED 11*, 1 - 5, September 2008, Riva del Garda, Trentino, Italy.