FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Assessing the Impact of Alternative Splicing in Thyroid Cancer

Luís Filipe Correia Cleto

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rui Camacho (FEUP)

Co-supervisor: Valdemar Máximo (I3S/IPATIMUP & FMUP)

Co-supervisor: Pedro Ferreira (I3S/IPATIMUP)

July 22, 2016

Assessing the Impact of Alternative Splicing in Thyroid Cancer

Luís Filipe Correia Cleto

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Eugénio Oliveira
External Examiner: Sérgio Matos
Supervisor: Rui Camacho

July 22, 2016

Abstract

All over the world there are millions of people affected by cancer. There are several possible causes for this disease, one of which is a genomic origin. While there are many alterations in cancer cells that distinguish them from healthy ones, there is also an abundance of aberrant mRNA transcripts in cancerous cells. These aberrant transcripts might lead to changes in the synthesized proteins which, in turn, will confer new, and possibly devastating, properties to the affected cell. One possible source of this defective mRNA is an abnormal gene splicing.

The advent of Next-Generation Sequencing (NGS) techniques has revolutionized the field of molecular biology in the past few years and brought many new approaches for cancer research. The goal of this thesis is to make use of these novel techniques, namely RNA-seq, in order to create a tool capable of detecting events of aberrant alternative splicing and assessing their impact on a cell or an organism's transcriptome. We have extended our initial goal and expanded our analysis with extra studies that are considered valuable by expert biologists in the study of cancer. The extra developments include the identification of fusion genes.

However, current tools of this type are commonly complex to use, require powerful computational resources and also need the user to be familiar with programming or other concepts from the field of informatics, which is often not the case for geneticists or molecular biologists. As such, this project also focuses on enabling these researchers to use the aforementioned tools through the creation of a distributed system which provides the computational resources necessary for an RNA-seq pipeline as well as a web interface to facilitate configuring and running experiments on this pipeline.

Lastly, the thesis proposal is evaluated on real data provided by IPATIMUP and also on data obtained from the ENCODE project.

Resumo

Por todo o mundo há milhões de pessoas afetadas pelo cancro. Existem múltiplas causas possíveis para esta doença, algumas das quais são uma origem génica. Enquanto que há muitas alterações em células cancerígenas que as distinguem das saudáveis, existe também uma abundância de registos de mARN aberrantes em células cancerígenas. Estes registos aberrantes irão levar a alterações nas proteínas sintetizadas que, por sua vez, poderão conferir às células propriedades novas, e potencialmente devastadoras. Uma possível origem deste mARN defeituoso é o splicing anormal de genes.

O advento das técnicas de sequenciação de nova geração tem revolucionado o campo da biologia molecular nos últimos anos e trouxe inúmeras novas abordagens para investigação do cancro. O objetivo desta tese é recorrer a estas novas técnicas, nomeadamente à técnica *RNA-seq*, de modo a criar uma ferramenta capaz de detetar eventos de splicing alternativo aberrante e avaliar o seu impacto no transcriptoma da célula ou do organismo. Conseguimos ainda ultrapassar os objetivos iniciais e incluir no estudo a deteção de genes de fusão, considerados muito relevantes para o aparecimento do cancro.

No entanto, ferramentas modernas deste tipo são frequentemente complexas de utilizar, necessitam de recursos computacionais poderosos e também exigem conhecimentos de programação e outros conceitos da área da informática por parte do utilizador, o que frequentemente não se verifica para geneticistas ou especialistas de biologia molecular. Como tal, este projeto também se foca em permitir a estes investigadores fazer uso das ferramentas mencionadas através da criação de um sistema distribuído que fornece os recursos computacionais necessários para uma pipeline de *RNA-seq* e também uma interface web para facilitar a configuração e execução de experiências nesta pipeline.

Finalmente, a proposta de tese é avaliada em dados reais fornecidos pelo IPATIMUP e também em dados obtidos do projeto ENCODE.

Contents

1 Introduction **1**
- 1.1 Problem Domain . . . 1
- 1.2 Motivation and Objectives . . . 2
- 1.3 Project . . . 3
- 1.4 Dissertation Structure . . . 3

2 Biological Key Concepts and Technology Survey **5**
- 2.1 Basic Biological Concepts . . . 5
  - 2.1.1 Glossary . . . 5
  - 2.1.2 Genetic Processes . . . 6
  - 2.1.3 Alternative Splicing . . . 7
- 2.2 RNA Sequencing and Transcriptome Assembly . . . 9
- 2.3 Relevant Standard File Formats . . . 12
- 2.4 Tools for RNA-seq . . . 15
  - 2.4.1 RNA-Seq Pipelines . . . 17
- 2.5 Data Repositories . . . 19
- 2.6 Additional Technologies . . . 19
- 2.7 Chapter Conclusions . . . 21

3 The eRAP Framework **23**
- 3.1 eRAP . . . 23
- 3.2 System Architecture . . . 24
  - 3.2.1 Data Scheme . . . 25
- 3.3 Use Cases . . . 27
- 3.4 Web Interface . . . 27
- 3.5 Analysis Server . . . 30
- 3.6 Case Studies . . . 31
  - 3.6.1 Standard Experiment . . . 31
  - 3.6.2 Experiment With Errors . . . 32
- 3.7 Chapter Conclusions . . . 32

4 RNA-seq Data Analysis **33**
- 4.1 Gene Fusion Analysis . . . 33
- 4.2 Alternative Splicing Analysis . . . 33
  - 4.2.1 Junctions Comparer . . . 34
  - 4.2.2 Usage . . . 34
  - 4.2.3 Junction Identification . . . 35
  - 4.2.4 Output . . . 36

4.2.5 Case Study . . . 37
4.3 Gene Expression Quantification . . . 38
4.3.1 Output . . . 38
4.4 Chapter Conclusions . . . 39

5 Case Study **41**
5.1 Samples characterization . . . 41
5.2 Research Objectives . . . 41
5.3 Analysis Protocol . . . 42
5.4 Genome version GRCh37 versus GRCh38 . . . 42
5.5 Results . . . 43
5.5.1 Gene Expression Analysis . . . 43
5.5.2 Alternative Splicing Analysis . . . 44
5.5.3 Gene Fusion Analysis . . . 44
5.6 Chapter Conclusions . . . 45

6 Conclusions and Future Work **47**
6.1 Conclusions . . . 47
6.2 Future Work . . . 48

References **49**

A R Analysis **53**
A.1 Dependencies . . . 53
A.2 Script Code . . . 53

B Experiment Results **57**
B.1 Gene Expression Results' Descriptions . . . 57
B.2 Alternative Splicing Results . . . 58
B.3 Alternative Splicing Results' Descriptions . . . 59

List of Figures

2.1 Simplified overview of the gene expression transcription and splicing processes and the structure of the genetic molecules involved . . . 7
2.2 Overview of the gene expression translation process and the structure of the genetic molecules involved . . . 8
2.3 The alternative splicing process. The colored blocks represent exons in the RNA sequence while the lines between them are introns . . . 9
2.4 Traditional classification of basic types of alternative splicing. In these graphics, exons are represented by boxes and introns by lines. Regions included in the messages by alternative splicing are yellow while constitutive exons are shown in blue . . . 10
2.5 Illustration of gene fusion events at a chromosomal level . . . 11
2.6 General RNA-seq workflow diagram . . . 12
2.7 iRAP pipeline architecture . . . 18

3.1 System Architecture . . . 24
3.2 System Use Cases . . . 28
3.3 Landing page of the UI module . . . 29
3.4 Experiments index of the UI module . . . 29
3.5 Experiment pages for a successful and a failed experiment . . . 30
3.6 Experiment creation page . . . 31

4.1 The graphic on the left shows the execution time for *Junctions Comparer* and MATS. The middle column includes the pre-processing time required to prepare the BAM samples for JC. The graphic on the right compares the results obtained by the two tools . . . 37

5.1 Most significant differentially spliced genes between wFTC and mFTC sample groups. Each graphic shows the normalized amount of alternative splicing events per sample for each gene. These results were obtained using the GRCh38 genome assembly . . . 45

LIST OF FIGURES

List of Tables

2.1 Tools integrated in the iRAP pipeline for each standard processing stage 18

3.1 MongoDB Collections 25
3.2 Variables to be declared in the *local_settings.py* file 28

4.1 Junctions Comparer Arguments 35
4.2 Junction Start Classification 36
4.3 Junction End Classification 36

5.1 Gene Expression Results for both sample groups using GRCh37 and GRCh38. The values for each column represent the average of the normalized fraction of reads that mapped to that gene in each sample 43
5.2 Gene Expression results for both sample groups using GRCh38. The values for the second and third column represent the average of the normalized fraction of reads that mapped to that gene in each sample. The last column represents the largest difference factor calculated as *MAX(wFTC/mFTC, mFTC/wFTC)* 44
5.3 Gene Fusion events detected in the original study and by EricScript using both GRCh37 and GRCh38. The gene symbols used to identify the fusion genes are the ones used in the older annotation for compatibility with the previous paper . . 46

B.1 Short description of genes that are differentially expressed between samples mFTC and wFTC 58
B.2 Alternative Splicing analysis results for both sample groups using GRCh38. The values for the third and fourth column represent the average of the normalized number of alternative splicing events that took place within that gene in each sample. The last column represents the largest difference factor calculated as *MAX(wFTC/mFTC, mFTC/wFTC)* 59
B.3 Short description of genes that show differential alternative splicing between samples mFTC and wFTC 61

LIST OF TABLES

Abbreviations

AS	Alternative Splicing
A3S	Alternative 3 Prime
A5S	Alternative 5 Prime
BAM	Binary Alignment/Map
BED	Browser Extensible Data
cDNA	Complementary DNA
DNA	Deoxyribonucleic Acid
ENCODE	Encyclopedia of DNA Elements
ES	Exon Skipping
FTP	File Transfer Protocol
GFF	Generic Feature Format
GLPv3	General Public License 3
GTF	General Transfer Format
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
I3S	Instituto de Investigação e Inovação em Saúde
IPATIMUP	Instituto de Patologia e Imunologia Molecular da Universidade do Porto
IR	Intron Retention
iRAP	Integrated RNA-seq Analysis Pipeline
MEE	Mutually Exclusive Exons
MATS	Multivariate Analysis of Transcriptome Splicing
mRNA	Messenger RNA
mtDNA	Mitochondrial DNA
NCBI	National Center for Biotechnology Information
NGS	Next Generation Sequencing
RNA	Ribonucleic Acid
RNA-seq	RNA sequencing
rRNA	Ribosomal RNA
SAM	Sequence Alignment/Map
SNP	Single Nucleotide Polymorphism
SSL	Secure Sockets Layer
TLS	Transport Layer Security
tRNA	Transport RNA
URL	Uniform Resource Locator
WTSS	Whole Transcriptome Shotgun Sequencing

Chapter 1

Introduction

In this chapter we will provide an overview of the context to which this thesis belongs. We will present our objectives with this work as well as the underlying motivations. A brief analysis of the developed project will be explained as well as the structure of the remainder of this report.

1.1 Problem Domain

Molecular biology is a branch of science that studies biological activity at the molecular level. It overlaps with biology and chemistry and in particular, genetics and biochemistry. A key area of this field concerns the understanding of how DNA, RNA and the synthesis of proteins function and how this affects the various cellular systems [Man14]. One of the processes studied by molecular biology is gene expression. During this process (further elaborated in Chapter 2), DNA sequences are translated into RNA that is used to synthesize proteins. Using this process DNA indirectly regulates the chemical activity of cells. Deregulation of gene expression is known to play a pivotal role in the origin of cancer [GYLL13].

One such possible deregulation is the compromise of splicing mechanisms, which allow a single gene to code different proteins. This will cause changes to the proteome (total set of proteins) of cells and possibly confer them new properties.

Next-Generation Sequencing (or *Massively Parallel Sequencing*) methods have been revolutionizing genomics and, in turn, genetics. Thanks to these methods, disciplines such as molecular biology are no longer restricted to small-scale genetic approaches but are able to introduce genome-wide thinking. As such, coupling these tools with the appropriate algorithms has the potential to radically alter our understanding of model organisms and ultimately of ourselves [Mar08].

One possible application of these instruments is in cancer research. While there have been many advances in our knowledge of cancer over the past decades, there is much that remains unknown about the mechanisms involved in its origin(s). The primary objective of this thesis is to resort to RNA-seq [WGS09], an NGS approach for transcriptome profiling, in order to create a tool capable of identifying and assessing the impact of aberrant alternative splicing events as a

possible origin for cancer. Additionally, it is also intended to measure the impact of gene fusion events for a more in-depth analysis of the alterations present in cancer cells.

However, current RNA-seq tools are commonly complex to use, require powerful computational resources and also need the user to be familiar with programming or other concepts from the field of informatics, which is often not the case for geneticists or molecular biologists. As it is intended to allow experts in the field (who may not have much programming knowledge) to conduct their own analysis through the use of this, and other tools, the creation of a user-friendly web-based application for interacting with the analysis system emerges as an additional objective. The application consists of a web interface that allows access to a distributed system which provides the computational resources necessary for an RNA-seq pipeline and a database where to archive results.

1.2 Motivation and Objectives

All over the world there are millions of people affected by cancer. There are several possible causes for this disease, one of which is a genomic origin. While there is a myriad of alterations in cancer cells that distinguish them from healthy ones, one trait that is common to all cancerous cells is an abundance of aberrant mRNA transcripts. One possible source of this defective mRNA is abnormal gene splicing. Alternative splicing is the regulated process during gene expression that results in a single gene coding for multiple proteins. A deregulation in normal splicing, caused by cancer-specific defects in the splicing mechanisms, can confer new properties of growth, differentiation or other characteristics to the cancer cell [FG08].

With the emergence of high-throughput sequencing there have been substantial advances in cancer genomics research. The goal of this project is the development of a tool, based on RNA-seq, that uses high-throughput information from the transcriptomes of thyroid cancer samples in order to identify events of aberrant splicing and assess their impact in the structure of the transcript. For this purpose, the RNA-seq technology was used for sequencing the genomes. RNA-seq performs the reconstruction of at least part of the genome of an individual through the reading of small fragments, calculates the sets of active and inactive genes and compares it with another reference genome (gene differentiation).

Further extensions to this tool have already been proposed such as focusing the analysis on mitochondrial DNA, as this is passed down from generation to generation without any modifications (except for rare mutations), making it more stable over time [ED]. It is also in the researchers' interests to incorporate information relative to events of gene fusion into the analysis to allow further insight into the modifications present in cancer cells.

However, it is common for tools of this type to require powerful resources and be complex to use, especially for molecular biologists or geneticists who might not have a programming background. As such, it is also relevant to create a web application to serve as an interface for a distribute system where an RNA-seq pipeline is deployed.

In a final stage to further assess the developed tool, we have used thyroid cancer data retrieved from I3S/IPATIMUP and also from the ENCODE project [Ins].

1.3 Project

This work focuses on the development of a prototype tool for detecting aberrant alternative splicing events and measure their impact on the organism by assessing the changes to the transcriptome induced by these events. This tool was used on real data provided by I3S/IPATIMUP and the ENCODE project in order to validate it and provide new insights on the origins of cancer. In order to generate more complete results, gene fusion and gene expression analyses were also conducted on the provided samples.

As we intend to enable the usage of RNA-seq tools of this type by researchers and experts in genomics without programming knowledge, the development of this project also encompasses the creation of a web interface for user interaction and a back-end that is capable of handling the analysis.

Analysing the transcriptome is a computationally intensive task. As such, the analysis module should be implemented in such a way that allows it to be separated from the rest of the system as it, ideally, will be running on a dedicated cluster with powerful machines. Our tool exposes a web API that allows executing and monitoring tasks as well as transferring data and results. In this system a pipeline for analysing data via RNA-seq will be deployed.

Another service was created for handling the submission of jobs to the cluster and for receiving and aggregating results, storing them in a persistent database. This service allows the interaction of a user with the system via a web interface so that they can submit jobs and view experiment reports in HTML format.

1.4 Dissertation Structure

In addition to this introductory chapter, the remainder of this report contains five additional chapters. In Chapter 2, a brief explanation of some key biological and RNA-seq concepts essential for understanding the subjects approached by this work is provided. Other relevant tools and technologies to be used with this project will also be detailed in this chapter.

In Chapter 3, we present the system proposed to enable molecular biologist and geneticists to make use of an RNA-seq pipeline: the eRAP system. We describe its architecture, detail each component/module's functions and describe the methodological approach for their implementation. The chapter concludes by characterizing the studies used to validate the tool.

Chapter 4 describes the types of analyses we aimed to perform on the data provided by IPATIMUP: gene fusion, alternative splicing and gene expression. For each analysis, the tools used and the necessary pre-processing steps were described in detail. Additionally, as the *Junctions Comparer* tool was entirely developed by us to perform the desired alternative splicing analysis, the case study used to validate this tool is also described in Section 4.2.5.

Chapter 5 introduces the case study we performed using real world thyroid cancer data provided by IPATIMUP's researchers. The chapter starts with the description of the samples provided, the researching objectives and the protocol of analysis. The following sections describe the results we obtained with our analysis.

Finally, Chapter 6 summarizes what has been described in this dissertation and presents the final conclusions. In this last chapter we also point out possibilities for future work and improvements/extensions for each of the developed tools.

Chapter 2

Biological Key Concepts and Technology Survey

In this chapter we present the basic biological concepts that are necessary to understand the thesis work. We begin by detailing the gene expression process, splicing and alternative splicing. This will be followed by a review of some of the different standard formats, tools and methods available for analysing RNA. We focus on the iRAP pipeline[1] and also on the state-of-the-art technologies necessary to implement the computational infrastructure and web portal.

2.1 Basic Biological Concepts

2.1.1 Glossary

We begin by listing some key terms and definitions that will be commonly used throughout the remainder of this dissertation:

- **DNA** - molecule that carries most of the genetic instructions used in the growth, development, functioning and reproduction of all known living organisms and many viruses.
- **chromosome** - a thread-like structure of nucleic acids and protein found in the nucleus of most living cells, carrying genetic information in the form of genes.
- **gene** - distinct sequence of DNA that forms part of chromosome.
- **RNA** - molecule that acts as a messenger carrying instructions from DNA for controlling the synthesis of proteins, although in some viruses RNA rather than DNA carries the genetic information.

[1] https://nunofonseca.github.io/irap/

- **exon** - a segment of a DNA or RNA molecule containing information coding for a protein or peptide sequence.
- **intron** - a segment of a DNA or RNA molecule which does not code for proteins and interrupts the sequence of genes.
- **genetic code** - the means by which DNA and RNA molecules carry genetic information in living cells.
- **codons** - a sequence of three nucleotides which together form a unit of genetic code in a DNA or RNA molecule.
- **aminoacids** - a simple organic compound which serves as a building block for proteins.
- **5'/3'** - pronounced 5 prime or 3 prime respectively. They indicate the carbon numbers in the DNA's sugar backbone. The 5' carbon has a phosphate group attached to it and the 3' carbon a hydroxyl group. This asymmetry gives a DNA strand a "direction". For example, DNA polymerase works in a 5' -> 3' direction, that is, it adds nucleotides to the 3' end of the molecule, thus advancing to that direction.
- **DNA strands** - the term *template strand* (or *sense strand*) refers to the sequence of DNA that is copied during the synthesis of mRNA. The opposite strand (that is, the strand with a base sequence directly corresponding to the mRNA sequence) is called the *coding strand* (or the *antisense strand*) because the sequence corresponds to the codons that are translated into protein.
- **fusion genes** - hybrid gene formed from two previously separate genes.

2.1.2 Genetic Processes

As mentioned in Chapter 1, gene expression is the process by which genes are "translated" into functional genetic products, such as RNA or, at a later stage, proteins, which ultimately regulate cellular activity. This process can be divided into two main phases:

- **Transcription** - The production of mRNA from DNA by the RNA polymerase enzyme, and further processing of the resulting mRNA molecule.
- **Translation** - The synthesis of proteins directed by mRNA, and subsequent processing of the resulting protein molecule.

It is also worth noting that not all genes are translated into proteins as a final product of gene expression. Some genes code different types of RNA that play a role in the translation phase such as tRNA or rRNA [GENb].

One of the processing phases that mRNA molecules undergo is called splicing. Simplifying the structure of genetic code, it can be divided into exons and introns. These sections of DNA

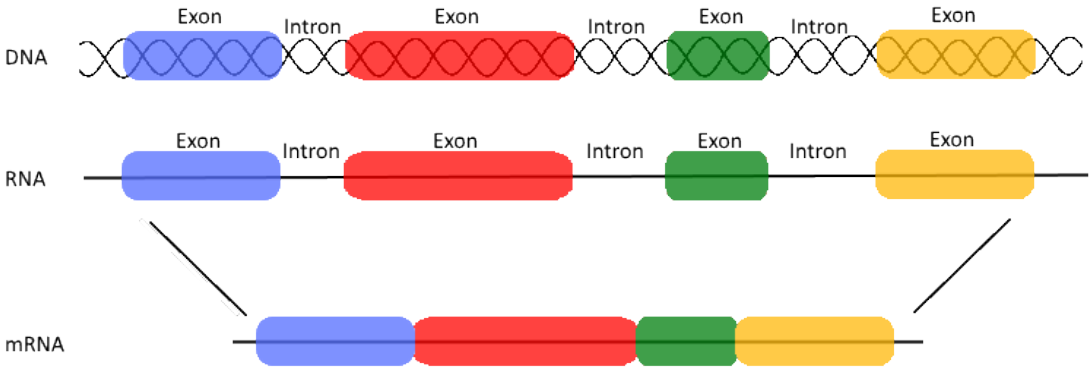

Figure 2.1: Simplified overview of the gene expression transcription and splicing processes and the structure of the genetic molecules involved

are kept in precursor mRNA (pre-mRNA) at first as it is initially transcribed from DNA. During the splicing phase, the introns are removed and the pre-mRNA is converted into mature mRNA containing only coding regions (exons), ready to be used for synthesizing proteins. This process is illustrated in Figure 2.1. During the protein synthesis phase, the RNA codons in the exons are "translated" into amino acids, forming chains that will eventually become proteins. This process, illustrated in Figure 2.2 starts with a special start codon (AUG) and ends when one of three stop codons is found (UAA, UAG or UGA).

However, it is also possible that during the splicing phase, some exons are also removed. This leads to a single pre-mRNA strand being capable of generating more than one distinct mature mRNA molecules, depending on which exons are kept in the final product. In turn, this implies that a single gene is able to code more than one protein. This process, shown in Figure 2.3, is known as alternative splicing.

It is possible for this mechanism to be compromised, inducing aberrant alternative splicing events [FG08]. This will lead to changes in the proteome: some proteins will no longer be synthesized or new ones will be added to the proteome. As proteins play a key role in regulating cellular activity, these changes may have a significant impact on the characteristics of the cancerous cells and confer them new and devastating properties such as capabilities of growth, differentiation and even hormone resistance [ASK+01].

2.1.3 Alternative Splicing

Pre-mRNA splicing is a natural source of cancer-causing errors in gene expression. Mutations in intronic splice sites of tumor suppressor genes often cause exon-skipping events which will result in the truncation of the synthesized proteins, just like classical nonsense mutations. It is also worth noting that many studies over the last 20 years have reported cancer-specific alternative splicing

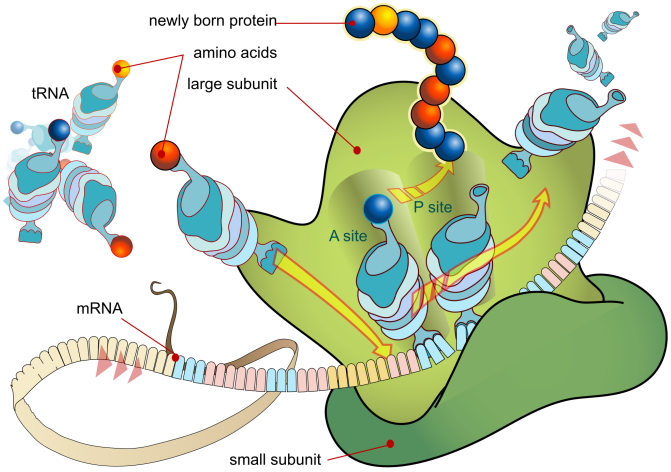

Figure 2.2: Overview of the gene expression translation process and the structure of the genetic molecules involved[2]

in the absence of genomic mutations [Ven04].

Types of Alternative Splicing

There are five basic modes of alternative splicing which are generally recognized [HKA]. As depicted in Figure 2.4, these modes can be classified as:

- **Exon skipping**: In this case, an exon may be spliced out of the primary transcript or retained. This is the most common mode in mammalian pre-mRNAs.
- **Mutually exclusive exons**: One of two exons is retained in mRNAs after splicing, but not both.
- **Alternative donor site**: An alternative 5' splice junction (donor site) is used, changing the 3' boundary of the upstream exon.
- **Alternative acceptor site**: An alternative 3' splice junction (acceptor site) is used, changing the 5' boundary of the downstream exon.
- **Intron retention**: A sequence may be spliced out as an intron or simply retained. This is distinguished from exon skipping because the retained sequence is not flanked by introns. If the retained intron is in the coding region, the intron must encode amino acids in frame with the neighboring exons, or a stop codon or a shift in the reading frame will cause the protein to be non-functional. This is the rarest mode in mammals.

[2]Image taken from: https://en.wikipedia.org/wiki/Translation_(biology)#/media/File:Ribosome_mRNA_translation_en.svg

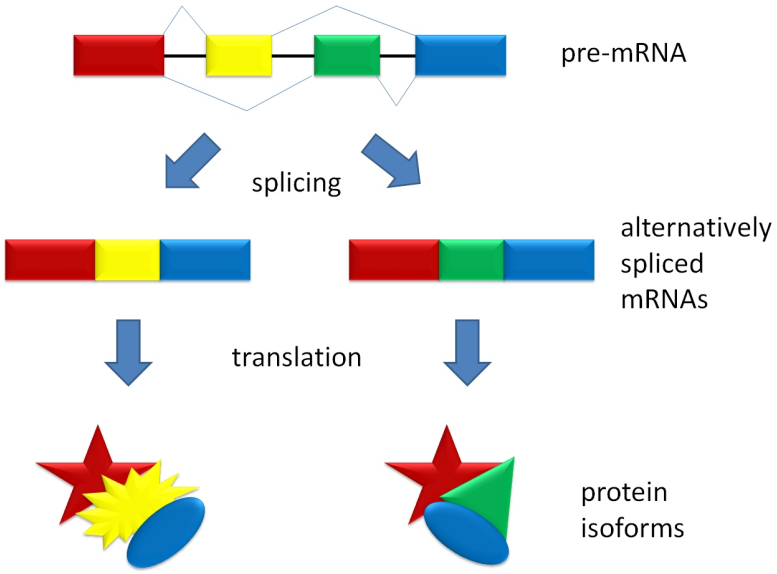

Figure 2.3: The alternative splicing process. The colored blocks represent exons in the RNA sequence while the lines between them are introns[3]

Fusion Genes

A fusion gene is a hybrid gene formed from two previously separate genes. As shown in Figure 2.5, it can occur as a result of: translocation, interstitial deletion, or chromosomal inversion. Currently, 358 gene fusions involving 337 different genes have been identified. Additionally, a growing number of gene fusion events are being recognized as important parameters for diagnosing (and prognosing) malignant hematological disorders and childhood sarcomas. The analysis shows that gene fusions occur in all malignancies, and that they account for 20% of human cancer morbidity [MJM07].

2.2 RNA Sequencing and Transcriptome Assembly

Understanding the transcriptome (set of all RNA molecules) is essential for interpreting the functional elements of the genome, revealing the molecular constituents of cells and tissues, and also for understanding development and disease. Obtaining information of a cell or organism's transcriptome is done experimentally, by employing sequencing techniques. These were initially dominated by Sanger's method, proposed in 1977, and other similar sequencing techniques [FCK02]. In spite of providing accurate results that came to revolutionize the fields of genetics, such methods were notably slow and costly: applying these techniques on large scale projects, such as the Human Genome Project, required the creation of sequencing centers that housed hundreds of DNA sequencing instruments which, in turn, required large amounts of personnel to operate [Sch07]. Due to the severity of the limitations imposed by these methods, this kind of study was mostly

[3]Image taken from: https://simple.wikipedia.org/wiki/Alternative_splicing

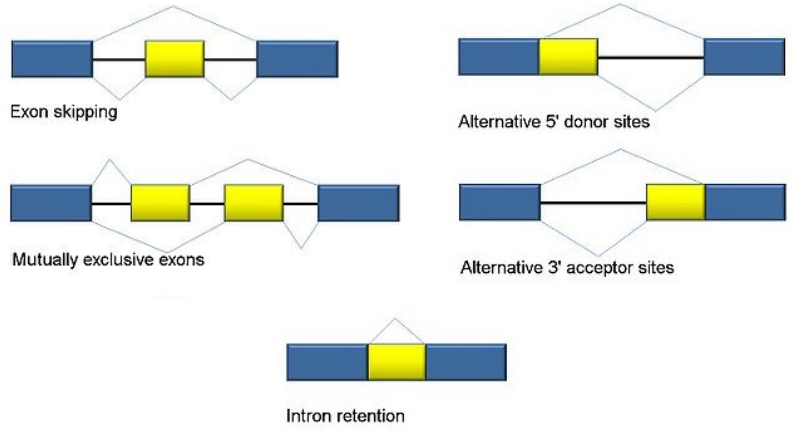

Figure 2.4: Traditional classification of basic types of alternative splicing. In these graphics, exons are represented by boxes and introns by lines. Regions included in the messages by alternative splicing are yellow while constitutive exons are shown in blue[4]

restricted to model organisms, such as the fruit fly and mouse genomes [Wol13]. It comes as no surprise that new techniques, capable of producing larger amounts of information at a lesser cost and being employed by single laboratories, have been rapidly growing in popularity.

For many years microarrays were the standard tool for quantifying gene expression and examining features of the transcriptome. However, this technique requires previous knowledge of the genome being studied. It also has propensity for introducing some biases in gene expression measurements from cross-hybridization. For these reasons, it is being replaced by a more recent approach called RNA-seq. Compared to microarrays, RNA-seq provides more informative and reliable results and is capable of applying *de novo* assembly approaches for exploratory analysis, discarding the need for a reference genome[5]. Further, it provides information on RNA splice events which are not readily detected by standard microarrays.

RNA-seq

The human genome contains 3,234.83 Mega-basepairs (Mbp) while that of *Escherichia Coli* contains only 4.6 million basepairs. This translates into roughly 700MB for humans if a perfect representation was possible (using only 2 bits per base). However, in reality, sequencing a genome requires a lot of information besides just the basepairs (such as quality information) and it becomes impossible to achieve this perfect representation. Sequencing a whole genome with an average x30 coverage (each base was covered, on average, by 30 reads) requires approximately

[4]Image adapted from: https://commons.wikimedia.org/wiki/File:Alt_splicing_bestiary2.jpg

[5]A reference genome is a digital nucleic acid sequence database, assembled by scientists as a representative example of a species' set of genes. *De novo* transcriptome assembly is the method of creating a transcriptome without the aid of a reference genome

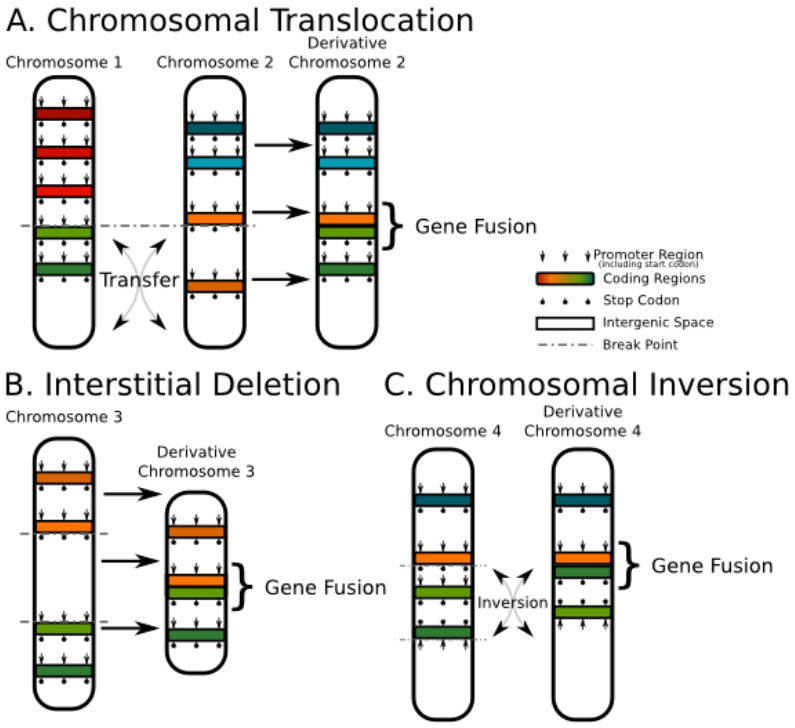

Figure 2.5: Illustration of gene fusion events at a chromosomal level[7]

90GB, mapping one base to one byte. Including quality scores and short reads[6], as the standard formats generally do, this can scale up to about 180GB of necessary storage [Rei]. These numbers show the complexity of the data being handled and the need for powerful computational resources and sophisticated algorithms for sequencing complex genomes such as the human genome.

For this thesis, we will focus on the RNA-seq approach for analysing the transcriptome. RNA-seq is a recently developed approach to transcriptome profiling that uses deep-sequencing technologies. It also provides a far more precise measurement of levels of transcripts and their isoforms than other methods. RNA-seq is a sequential process that is still under development [WGS09]. A workflow diagram of common steps in the RNA-seq analysis is depicted in Figure 2.6.

From analysing the diagram, several stages of the process can be identified. Initially, short reads are obtained by sequencing cDNA[8] fragments (reads) obtained from a population of RNA. These reads are typically subjected to a **quality control** stage in which reads below a certain quality threshold are discarded.

Following quality control, the resulting reads are either **aligned** to a reference genome or reference transcripts. If no reference genome is provided, a ***de novo* assembly** is performed to

[6]The term 'short read' came about to describe technologies that produced reads that were substantially shorter (30-50bp) than the mainstream technologies employed at that point (1000bp)

[7]Image taken from: https://en.wikipedia.org/wiki/Fusion_gene

[8]complementary DNA (cDNA) is double-stranded DNA synthesized from a single stranded RNA template in a reaction catalysed by the enzyme reverse transcriptase

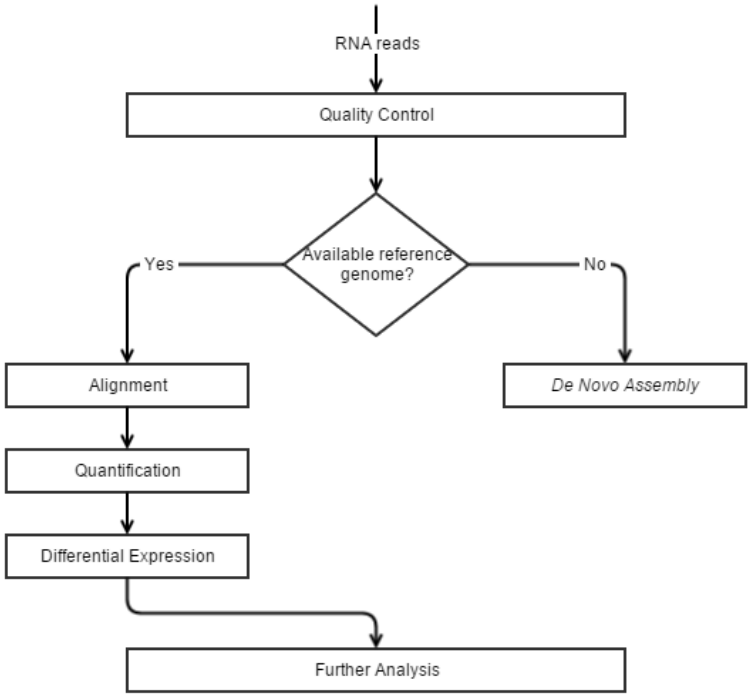

Figure 2.6: General RNA-seq workflow diagram

produce a genome-scale transcription map that consists of both the transcriptional structure and level of expression for each gene.

After the alignment stage, a **quantification** is performed in order to summarize and aggregate the reads over a biologically meaningful unit such as exons, isoforms or genes. By analysing the output of the latter phase it is possible to determine which genes are being expressed through **differential expression**. In turn, this allows determining which proteins will be synthesized. The analysis can be further extended with other optional steps such as gene set enrichment or splicing detection.

2.3 Relevant Standard File Formats

As there is a multitude of sequencing tools available, it comes as no surprise that there are also several distinct file formats used by these tools. In order to be able to integrate the sequencing tools, it becomes necessary to operate with several of these file types. We now briefly describe the file formats relevant to our work.

FASTA

FASTA is the line and character sequence format used by the National Center for Biotechnology Information (NCBI), resorting to their own character code conventions. It is a simple

text format, that can be used to store either nucleotide (DNA, RNA) or amino acid (protein) sequences [NCB]. Due to its simplicity and ease of parsing and manipulating, this format is broadly used to store sequencing reads. It is also an attractive solution for data transfer between different tools as several bioinformatic tools use it. An example can be seen in Listing 2.1.

```
1 >gi|31563518|ref|NP_852610.1| microtubule-associated proteins 1A/1B light chain 3A
      isoform b [Homo sapiens]
2 MKMRFFSSPCGKAAVDPADRCKEVQQIRDQHPSKIPVIIERYKGEKQLPVLDKTKFLVPDHVNMSELVKI
3 IRRRLQLNPTQAFFLLVNQHSMVSVSTPIADIYEQEKDEDGFLYMVYASQETFGFIRENE
```

Listing 2.1: A simple example of one sequence in FASTA format. The first line describes the sequence while the following lines contain its data

FASTQ

The FASTQ format has emerged as another common format to store biological sequences. It provides an extension to the FASTA format in that it can also store a numeric quality score associated with each nucleotide in a sequence [CFG+10]. It was originally developed by the Wellcome Trust Sanger Institute to bundle a FASTA sequence and its quality data, but has become the *de facto* standard for storing the output of high-throughput sequencing instruments. While this format suffered from the lack of a formal definition, the Sanger version of the FASTQ format has found the broadest acceptance. An example of a FASTQ file fragment is shown in Listing 2.2.

```
1 @SEQ_ID
2 GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
3 +
4 !''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

Listing 2.2: A simple example of one sequence in FASTQ format. The first line begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line). Line 2 contains the sequence data. Line 3 starts with a '+' sign and an optional identifier and any description. Line 4 contains the quality values for line 2's sequence.

SAM and BAM

The Sequence Alignment/Map (SAM) format is a generic alignment format for storing read alignments against reference sequences. It supports short and long reads from different sequencing tools and it is flexible, compact, and efficient in random access. The Binary Alignment/Map (BAM) format contains the same information as SAM but in binary format rather than in textual representation. It was designed in order to improve performance [LHW+09]. If present, the header must appear prior to the alignments. Header lines start with '@', while alignment lines do not. Each alignment line has 11 mandatory fields for essential alignment information such as mapping

position, and variable number of optional fields for flexible or aligner specific information. An example of a SAM file can be seen in Listing 2.3.

```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:45
r001 99 ref 7 30 8M2I4M1D3M = 37 39 TTAGATAAAGGATACTG *
r002 0 ref 9 30 3S6M1P1I4M * 0 0 AAAAGATAAGGATA *
r003 0 ref 9 30 5S6M * 0 0 GCCTAAGCTAA * SA:Z:ref,29,-,6H5M,17,0;
r004 0 ref 16 30 6M14N5M * 0 0 ATAGCTTCAGC *
r003 2064 ref 29 17 6H5M * 0 0 TAGGC * SA:Z:ref,9,+,5S6M,30,1;
r001 147 ref 37 30 9M = 7 -39 CAGCGGCAT * NM:i:1
```

Listing 2.3: Example of an alignment in SAM format. The first line ('@HD') is the header line and must contain the format version (VN). The '@SQ' lines identifies the reference sequence. Any other lines that don't begin with '@' represent alignments and have 11 mandatory information fields

GFF and GTF

The Generic Feature Format (GFF) is a standard file format for storing genomic features in a text file. GFF files are plain text, 9 column, tab-delimited files and they can also be used for databases with a custom schema [Gena]. GFF has several versions, the most recent of which is GFF3. The GFF format requires fields to be tab-delimited and only the last field of a feature line is allowed to not have a value. This format incorporates fields for identifying the feature, respective chromosome, the program that generated it and other useful information. An example can be seen in Listing 2.4.

The General Transfer Format (GTF) is quite similar to GFF (sometimes being referred to as GFF2.5), which it extends by defining additional conventions.

```
X Ensembl Repeat  2419108 2419128 42      .      .  hid=trf; hstart=1; hend=21
X Ensembl Repeat  2419108 2419410 2502    -      .  hid=AluSx; hstart=1; hend=303
X Ensembl Pred.trans.     2416676 2418760 450.19 -  2      genscan=GENSCAN019335
X Ensembl Variation       2413425 2413425 .      +  .
X Ensembl Variation       2413805 2413805 .      +  .
```

Listing 2.4: Example of GFF output from an Ensembl export. Each line represents a feature and contains 9 fields respectively identifying the chromosome/scaffold, program source, feature type, start and end positions, score, strand, frame and attributes (semi-colon separated list)

BED

BED (Browser Extensible Data) [gro] format provides a flexible way to define the data lines that are displayed in an annotation track. BED lines have three required fields and nine additional optional fields. The number of fields per line must be consistent throughout any single set of data in an annotation track. The order of the optional fields is binding: lower-numbered fields must always be populated if higher-numbered fields are used. An example of a BED file can be seen in Listing 2.5.

```
1 track name=pairedReads description="Clone Paired Reads" useScore=1
2 chr22 1000 5000 cloneA 960 + 1000 5000 0 2 567,488, 0,3512
3 chr22 2000 6000 cloneB 900 - 2000 6000 0 2 433,399, 0,3601
```

Listing 2.5: Example of an annotation track that uses a complete BED definition. The header lines begin with the word "browser" or "track" to aid the browser in the display and interpretation of the lines of BED data. The data lines contain 3 mandatory arguments (the first 3) which identify the chromosome/scaffold, start and end position respectively. The 9 additional fields are optional and identify the following features in this order: the name of the line, score, strand, thick start, thick end, RGB value (for the browser), block count, block sizes and block starts

2.4 Tools for RNA-seq

As aforementioned, there are many tools available for each processing stage of the RNA-seq analysis. In this section we will briefly explain some of the tools which are relevant to this thesis. Many of these are included in the iRAP pipeline described in Section 2.4.1.

Bowtie

Bowtie[9] is a short read aligner. Bowtie aligns reads with the aid of an index of the reference genome and it can also take advantage of multiple processor cores, if available, to achieve speedups and memory-efficiency. The most recent version of this software is Bowtie 2 which improves in speed, sensitivity and accuracy [LS12]. This tool is most effective when aligning short read sequences against large reference genomes.

TopHat

TopHat [TPS09] is a fast splice junction mapper for RNA-Seq reads. It uses Bowtie for aligning the short reads and then analyzes the mapping results to identify splice junctions between

[9]Bowtie user manual: http://bowtie-bio.sourceforge.net/manual.shtml

exons. It can take advantage of multiple processor cores by using parallel processes. These features make it much faster than previous systems, mapping nearly 2.2 million reads per CPU hour, which is enough to process the data from a whole RNA-Seq experiment in under a day on a standard desktop computer.

Cufflinks

Cufflinks [TRG+12] is commonly used together with TopHat and Bowtie as it is part of the Tuxedo Suite tools. It assembles transcripts, estimates their abundances, and tests for differential expression and regulation in RNA-Seq samples. It receives the already aligned RNA-Seq reads as input and assembles the alignments into a set of transcripts. Cufflinks then uses the supporting reads to estimate the relative abundances of the transcripts, taking into account biases in library[10] preparation protocols.

HTSeq

HTSeq [APH14] is a Python library to facilitate the rapid development of scripts for processing high-throughput sequencing data. It provides parsers for many common data formats in these types of projects, as well as classes to represent data (genomic coordinates, sequences, sequencing reads, alignments, gene model information and variant calls) and provides data structures that allow for querying via genomic coordinates. The HTSeq-count tool was developed with this library for preprocessing RNA-Seq data for differential expression analysis by counting the overlap of reads with genes.

FastQC

FASTQC [A+10] provides a simple way to do quality control checks on raw sequence data coming from high throughput sequencing pipelines. It accepts data in BAM, SAM or FASTQ format and quickly identify possibly problematic areas. It also allows the summarization of the data with graphs and HTML reports.

SAMTools

SAMTools [LHW+09] is a library and software package for parsing and manipulating alignments in the SAM or BAM formats. It enables converting from different alignment formats, sorting and merging alignments, removing polymerase chain reaction (PCR) duplicates, generating per-position information in the pileup format, calling single nucleotide polymorphisms (SNPs)

[10]The sequencing library is prepared by random fragmentation of the DNA or CDNA sample, followed by 5' and 3' adapter ligation. Alternatively, "tagmentation" combines the fragmentation and ligation reactions into a single step that greatly increases the efficiency of the library preparation process. Adapter-ligated fragments are then PCR amplified and gel purified.

and short indel[11] variants, and showing alignments in a text-based viewer.

BEDTools

BEDTools [QH10] is a collection of utilities for a wide-range of genomics analysis tasks. The most widely-used tools enable genome arithmetic: intersect, merge, count, complement, and shuffle genomic intervals from multiple files in widely-used genomic file formats such as BAM, BED, GFF/GTF and the Variant Call Format (VCF). While each individual tool is designed to do a relatively simple task (e.g., intersect two interval files), quite sophisticated analyses can be conducted by combining multiple bedtools operations on the UNIX command line.

MATS

The Multivariate Analysis of Transcript Splicing (MATS) tool is used to detect differential alternative splicing events from RNA-Seq data between two samples [SPL+14]. From the RNA-Seq data, MATS can automatically detect and analyze alternative splicing events corresponding to all major types of alternative splicing patterns.

deFuse

The deFuse [MHZ+11] software is a computational method for fusion genes discovery in tumor RNA-seq data. It uses a dynamic programming-based split read analysis with discordant paired end alignments and does not discard paired end reads that align ambiguously. Additionally, deFuse is not limited to finding gene fusions with boundaries between known exons: it can identify fusion boundaries in the middle of exons or involving intronic or intergenic sequences.

EricScript

EricScript [BPM+12] is a computational framework used for the discovery of gene fusion events in RNA-seq data. Using the EricScript simulator, it is able to generate synthetic gene fusions and it can also determine statistical measures for evaluating gene fusion detection methods' performance by using the EricScript CalcStats tool. Comparatively to other similar tools it is a lightweight framework with high sensitivity [KVQL16], yielding accurate results with minimal memory and time consumption.

2.4.1 RNA-Seq Pipelines

While RNA-seq has become the technique of choice for whole-transcriptome profiling, applying these techniques requires considerable programming skills and computational resources. It is

[11] Molecular biology term for the insertion or the deletion of bases in the DNA of an organism

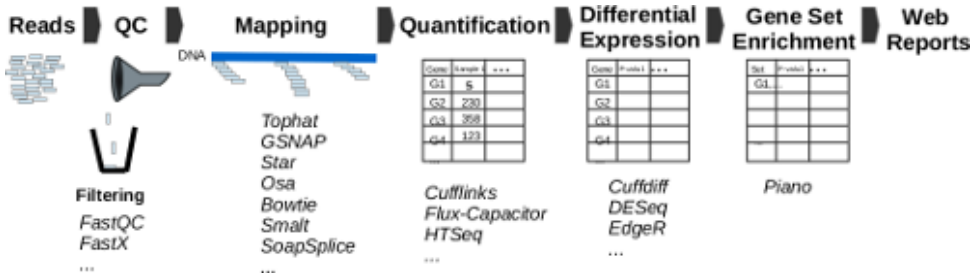

Figure 2.7: iRAP pipeline architecture [FPMB14]

necessary to process millions of sequencing reads and, at each step of the processing pipeline, there are multiple tools available each with their own advantages and disadvantages. This makes it common to require a specific combination of tools for each different experiment. However, integrating the desired tools can also be an arduous and lengthy task as the input/output files of each tool may not be compatible.

The iRAP [FPMB14] system (available under the GLPv3 license) is an integrated RNA-seq analysis pipeline that integrates many of the major tools available for each step of the RNA-seq process. It targets a broad audience and allows users with little bioinformatics knowledge to customize the pipeline and apply their tools of choice. However, more advanced users are also able to fully customize the options used by the chosen tools. The iRAP pipeline is depicted in Figure 2.7 as a standard analysis pipeline. Table 2.1 lists the available tools for each of the most common stages of an RNA-seq analysis (iRAP incorporates several other tools for other less common processing steps as well).

The iRAP pipeline accepts FASTQ or BAM files as input with raw sequence reads, a reference genome in FASTA format, gene model annotation in GTF format and a configuration file. iRAP then generates web reports that the user can read to explore the results of the analysis.

Table 2.1: Tools integrated in the iRAP pipeline for each standard processing stage

Stage	**Tools Supported**
Quality Control	*fastqc, fastx, bowtie1*
Mapping	*tophat1, tophat2, smalt, gsnap, soapsplice, bwa1, bwa2, bowtie1, bowtie2, gem, star, osa, mapsplice*
Quantification	*basic, htseq1, htseq2, cufflinks1, cufflinks2, cufflinks1_nd, cufflinks2_nd, scripture, flux_cap, nurd, stringtie, stringtie_nd, rsem, kallisto*
Differential Expression	*cuffdiff1, cuffdiff2, cuffdiff1_nd, cuffdiff2_nd, deseq, edger, voom, deseq2*
Gene Set Enrichment	*piano*
Gene Fusion Analysis	*fusionmap*

2.5 Data Repositories

In order to evaluate the system to be developed, it is necessary to have access to real data. Apart from the data that will be provided by IPATIMUP[12], there are several online repositories available online that provide such data as well.

One such repository that we will resort to for obtaining test data is the ENCODE project[13]. The goal of ENCODE is to build a comprehensive list of all functional elements in the human genome. This includes elements that act at the protein and RNA levels, and regulatory elements that control cells and circumstances in which a gene is active. Through their platform it is possible to access and download several datasets that can be used to validate our thesis work.

Other useful repositories and tools with genomic information also exist such as the ENSEMBL project[14] and the UCSC Genome Browser[15]. Both of these allow downloading data via FTP. Additionally, the Cancer Genome Atlas (CGA) provides data on cancer genome sequences, alignments and mutations for several cancer samples on the Cancer Genomics Hub[16]. One more interesting repository is that of the Genotype-Tissue Expression project[17], which collects and analyzes multiple human tissues from donors and allows downloading samples and other data such as RNA-seq results.

2.6 Additional Technologies

While the work to be done in this thesis focuses mainly on extending the RNA-seq analysis to detect and assess aberrant alternative splicing events, it also encompasses the creation of web services and a website to allow users to run the analysis, maintaining a database with the relevant results. To this end several other technologies are necessary.

AWK

Awk [AKW78] is a unix command encorporating a powerful "programming language" for file analysis and filtering. Its basic operation is to search a set of files for patterns, and to perform specified actions upon lines or fields of lines which contain instances of those patterns. Awk patterns may include arbitrary boolean combinations of regular expressions and of relational operators on strings, numbers, fields, variables, and array elements. Actions may include the same pattern-matching constructions as in patterns, as well as arithmetic and string expressions and assignments, if-else, while, for statements, and multiple output streams.

[12] https://www.ipatimup.pt/Site/
[13] https://www.encodeproject.org/
[14] http://www.ensembl.org/index.html
[15] https://genome.ucsc.edu/
[16] https://cghub.ucsc.edu/
[17] http://www.gtexportal.org/home/

Awk makes certain data selection and transformation operations easy to express, making it an ideal utility for quickly filtering and processing text-based files such as the ones commonly used in the bioinformatics field for describing genomes or sequencing reads.

Python

For the development of the system we decided to use the Python[18] language. It is a widely used general-purpose, high-level programming language that emphasizes code readability. Many Python libraries are freely available containing helpful modules for tasks such as web development or even biological computations (such as HTSeq or Biopython[19]).

Django

Django[20] is a high-level Python Web framework. The framework includes features such as authentication, URL routing, a templating system, an object-relational mapper (ORM), and database schema migrations. It follows the Model-View-Controller architectural pattern and has been used in several popular sites such as Pinterest[21] and Instagram[22].

There are alternatives to Django such as Flask, Pylons, Grok and TurboGears. Django was chosen as it is a very mature framework with sane defaults which allows developers to dive in to web applications quickly without needing to make a lot of decisions about their application's infrastructure ahead of time [Bro15]. This makes it ideal for mid-size and large projects. It is also the framework with the largest and most active community.

Databases

For the database it was decided to use a non-relational database as we will be storing massive volumes of data with a possibly different format between experiments. As such, this type of databases provide a flexibility that their relational counterparts do not. It might also become relevant at some point to store the data in a distributed manner across several machines, which is better handled by non-relational databases [LM13].

The non-relational database we opted to use was MongoDB[23]. It is an open source, document-oriented database designed with both scalability and developer agility in mind. Instead of storing your data in tables and rows as you would with a relational database, in MongoDB you store JSON-like documents with dynamic schemas. Additionally, MongoDB provides the GridFS system[24] which is a specification for storing and retrieving files that exceed the BSON-document size limit of 16 MB. Instead of storing a file in a single document, GridFS divides the file into parts, or

[18] https://www.python.org/
[19] http://biopython.org/wiki/Main_Page
[20] https://www.djangoproject.com/
[21] https://pinterest.com
[22] https://www.instagram.com/
[23] https://www.mongodb.org/
[24] https://docs.mongodb.com/manual/core/gridfs/

chunks, and stores each chunk as a separate document. GridFS is useful not only for storing files that exceed 16 MB but also for storing any files for which you want access without having to load the entire file into memory, making it useful for the project as the size of both input and output files for RNA-seq analysis tends to be in the order of a few gigabytes.

Web Services

A Web service is a service offered by an electronic device to another electronic device where communication takes place over the World Wide Web (WWW)[25]. Generally, a web service provides an object-oriented web based interface to a database server, which is utilized for example by another web server or mobile application that provide a user interface to the end user. It is also common for web servers to consume several web services at once on different machines in order to compile the content into a single user interface.

For implementing the web services we resorted to Python and Django, using the latter's templating language to build the website. To simplify the development of these modules, the PyCharm[26] integrated development environment was used.

2.7 Chapter Conclusions

In this chapter we provided an overview of the key molecular biology concepts that are essential for the understanding of the thesis work. We also detailed the RNA-seq process and presented short analyses of concrete tools and standard file formats that are relevant to the project. Finally, we also gave a brief introduction to the other technologies used for the project.

[25] https://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#webservice
[26] https://www.jetbrains.com/pycharm/

Chapter 3

The eRAP Framework

Let us recall that one of the problems we propose to solve is the development of a tool that helps experts to run RNA-seq analysis without requiring any programming knowledge or familiarity with a command-line interface. This tool is able to use powerful computational resources, it is based on an existing RNA-seq pipeline (iRAP) and has a user-friendly web interface. In this chapter we first introduce the architecture of our framework, the *Easy RNA-seq Analysis Pipeline*, and then detail each of its components as well as the system's usage.

3.1 eRAP

Due to the complexity of configuring an iRAP experiment, a user friendly web interface was designed to allow the creation of said experiments and the retrieval of results. The experiment creation form requires as input the main settings necessary to create an iRAP experiment such as the experiment's (unique) name, species name and reads in FASTA or FASTQ formats. It is not limited to any specific species as it allows users to specify any reference files (annotation in GTF format and reference genome in FASTA format) by indicating the URLs where they are available. The reference files are then retrieved from the specified URLs and kept locally to allow re-use across experiments and avoid wasting time and storage. However, as there are dozens of possible tools to use, the configuration options are also very vast and many have not been implemented on the interface at this stage. These additional options should be specified on the configuration file[1] which should be submitted when creating an experiment.

After creating an experiment, it becomes available on the global experiment list page as well as on the owner's list of experiments. On these pages the status of the experiments is available and further details can be seen on each experiment's specific page. There the results, output and error logs will also be available once an experiment finishes running.

[1] https://github.com/nunofonseca/irap/wiki/8-Configuration-file

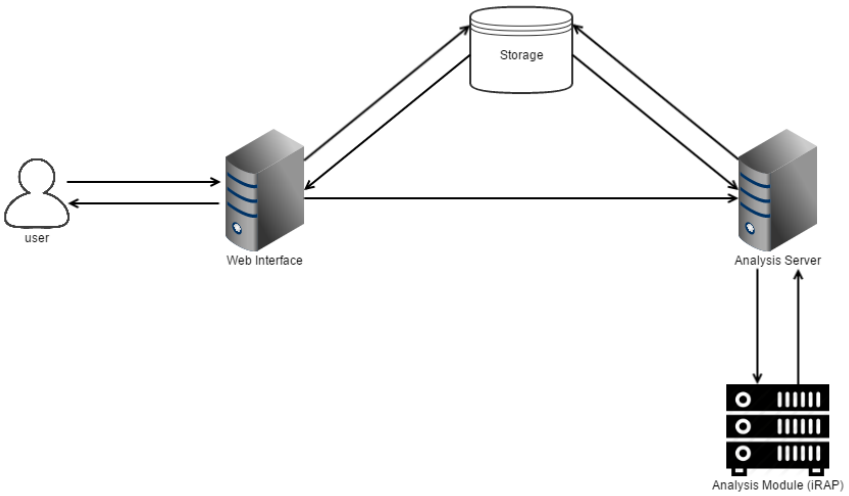

Figure 3.1: System Architecture

3.2 System Architecture

The system we implemented to answer the aforementioned problem has its foundation on the iRAP system. As depicted in Figure 3.1, there is an independent module responsible for the RNA-seq analysis, which will be a deployment of iRAP. As this component will be responsible for the computationally intensive operations, it will be deployed in a dedicated cluster with sophisticated machines. In this subsystem is included an additional back-end server to allow interaction with the front-end server and the database and it will be through this server that tasks will be created for iRAP.

In order to allow user interaction and for aggregating experiment data, the back-end module will expose an API, via Python with Django, to allow submitting and monitoring tasks. Additionally, it will resort to parallelization by maintaining a pool of worker processes in order to allow running multiple tasks at once. Once it receives a request to start an experiment, it will retrieve that experiment's files, which are stored in the database, and the experiment process will be added to the queue and remain there until a worker process is free and can start processing the experiment. The results will then be processed and stored in a MongoDB database which will be accessed whenever the status of the experiment is updated or results are finished. The output and error logs of the experiment's execution will also be stored in the database and made available to the users. The front-end server will host a web application in order to provide a user friendly interface and allow researchers without extensive programming knowledge to use the system to run their experiments and analyse the results.

Thanks to the decoupling of website, storage and analysis module, it will be simple to change the configuration of the computational resources at any time.

Table 3.1: MongoDB Collections

Stage	Tools Supported
auth_user	User account data
django_session	Session data for active users
storage.chunks	GridFS field where files' data is stored
storage.files	GridFS field where files' metadata is stored
user_server_experiments	Experiment data
user_server_gtffile	URL and GridFS object for GTF files
user_server_refgenome	URL and GridFS object for reference genome files

3.2.1 Data Scheme

MongoDB is a NoSQL, document based, database. The information is organized in collections in which each element is stored as a JSON document. MongoDB is also capable of storing and serving very large files without having to load them entirely into memory via the GridFS system, which splits files into chunks and saves each one as a separate document. Table 3.1 shows the existing collections used for storing the system's data.

Auth User

An Auth User's document stores all relevant information about a user: a unique ID, username, first and last names, email, password, type of user and other information such as the last login and the join date. This collection is created by the non-relational Django authentication back-end but is easily extendable by developers if necessary. Listing 3.1 shows an example of a document in this collection.

```
{
        "_id" : ObjectId("5739ee7f8bc3b46ad3fab34a"),
        "username" : "luiscleto",
        "first_name" : "Luis",
        "last_name" : "Cleto",
        "is_active" : true,
        "email" : "luisccleto@gmail.com",
        "is_superuser" : true,
        "is_staff" : true,
        "last_login" : ISODate("2016-05-29T20:15:32.397Z"),
        "password" : "pbkdf2_sha256$12000$ofxfyaH1uA7S$5TMWvOlCwIGVh4sviT2+
            XBz7thyKkZKQbVi/gOHYl+k=",
        "date_joined" : ISODate("2016-05-16T15:59:59.765Z")
}
```

Listing 3.1: Example of an Auth User document in JSON format

Django Session

In addition to an unique ID, This collection stores information about users' sessions such as the expiration date and encrypted session data. It is automatically created by Django's authentication back-end. An example of a Django session document can be seen in Listing 3.2.

```
{
        "_id" : "wh2bvr4cehb4l5lgm8hteousfdotyrlj",
        "expire_date" : ISODate("2016-06-06T14:01:40.873Z"),
        "session_data" : "
            OWQ1NGFlZmJhMzQxNTVjZjcyYTdiYWUwZGU5MzJmYjE2ODBiN2VmNjp7fQ=="
}
```

Listing 3.2: Example of a Django Session document in JSON format

Storange Chunks and Storage Files

Storage Chunks only stores binary data, the sequence number of the file's chunk and the ID of the corresponding file. Storage files on the other hand stores all relevant metadata about a file such as file size, chunk size, file name, upload date and an MD5 checksum hash[2]. These collections are automatically created by the GridFS system and their documentation is openly available[3].

User Server Experiments

This collection stores all data about an experiment such as its name and files. Additionally, it also maintains a status value which indicated the rate of completion of the experiment (a negative value indicates an error). An example of an experiment document is shown in Listing 3.3.

```
{
        "_id" : ObjectId("574b664f8bc3b44e50b614c1"),
        "status" : 100,
        "libraries_file" : "/ecoli_libraries.zip",
        "date_modified" : ISODate("2016-05-29T21:59:34.690Z"),
        "description" : "testing",
        "title" : "ecoli_ex",
        "gtf_file_id" : ObjectId("574b66468bc3b44e50b61017"),
        "reference_genome_id" : ObjectId("574b66468bc3b44e50b61016"),
        "author" : "luiscleto",
        "conf_file" : "/ecoli_example.conf",
        "err_log" : "/ecoli_ex.err_IKB20dS.log",
```

[2]The MD5 algorithm is a widely used hash function producing a 128-bit hash value. It is often used as a checksum to verify data integrity, but only against unintentional corruption.

[3]https://docs.mongodb.com/manual/core/gridfs/#gridfs-collections

```
        "fail_message" : "",
        "results_archive" : "/report.zip",
        "out_log" : "/ecoli_ex.out_Ep8yHb8.log",
        "date_created" : ISODate("2016-05-29T21:59:34.690Z"),
        "species" : "ecoli_k12"
}
```

Listing 3.3: Example of an Experiment document in JSON format

User Server GTFFile and RefGenome

Both collections have the same data scheme and contain a file's URL and content. The *file_content* field indicates the respective GridFS field if the file has already been retrieved and stored. Listing 3.4 shows an example of a GTFFile document in JSON format.

```
{
        "_id" : ObjectId("574b66468bc3b44e50b61017"),
        "file_content" : "/0f2cecc6-1625-4919-aefe-59a733d8dd7f.gz",
        "date_created" : ISODate("2016-05-29T21:59:34.687Z"),
        "file_address" : "ftp://ftp.ensemblgenomes.org/pub/release-19/bacteria//gtf
            /bacteria_22_collection/escherichia_coli_str_k_12_substr_mg1655/
            Escherichia_coli_str_k_12_substr_mg1655.GCA_000005845.1.19.gtf.gz"
}
```

Listing 3.4: Example of a GTF File document in JSON format

3.3 Use Cases

The eRAP system was designed with the researcher in mind as the end user. As Figure 3.2 shows, a user is able to perform basic authentication tasks such as registering or logging in or out of his account. Additionally, they can create an experiment and also view that experiment's status and retrieve any input or output files.

3.4 Web Interface

The implementation of the front-end Web Interface module was done using Python with Django and the Jinja2[4] templating language. Additionally, the Bootstrap framework[5] was used to simplify designing the application. In order for the server to be deployed successfully, the python requirements in the *requirements.txt* file provided with the source code must be installed. Additionally, the *non-rel* version of django should be installed from the *django-nonrel* project. Finally,

[4] http://jinja.pocoo.org/docs/dev/
[5] http://getbootstrap.com/

Figure 3.2: System Use Cases

before running the server, a *local_settings.py* file should be created in the irap_user_server directory which declares the variables shown in Table 3.2. Note that the last three variables are only relevant for the Analysis server and are not used for the Web Interface server.

Table 3.2: Variables to be declared in the *local_settings.py* file

Variable	**Description**
EMAIL_HOST_USER	email address to be used for SMTP authentication
EMAIL_HOST_PASSWORD	password to be used for SMTP authentication
IRAP_SERVER_ADDRESS	address of the back-end (analysis) server
FRONT_END_SERVER_ADDRESS	address of the front-end (web interface) server
MAX_NUMBER_OF_PROCESSES	size of back-end server's worker process pool
IRAP_DIR	work directory of the iRAP program, should be the same as the $IRAP_DIR environment variable

Once a user accesses the website, he is greeted with the main page (Figure 3.3) from where he has the option to perform account specific actions (register, login or logout) through menus in the top bar, or browse or create experiments through the menus in the left bar.

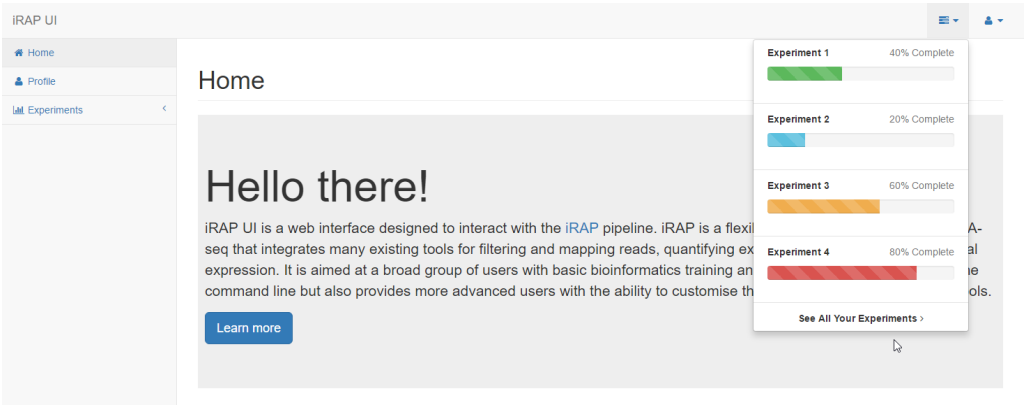

Figure 3.3: Landing page of the UI module

On the experiments index, a user can either see another user's or all experiments that are on the system. On this page the title of the experiment is shown along with a progress and status indicator, as shown on Figure 3.4.

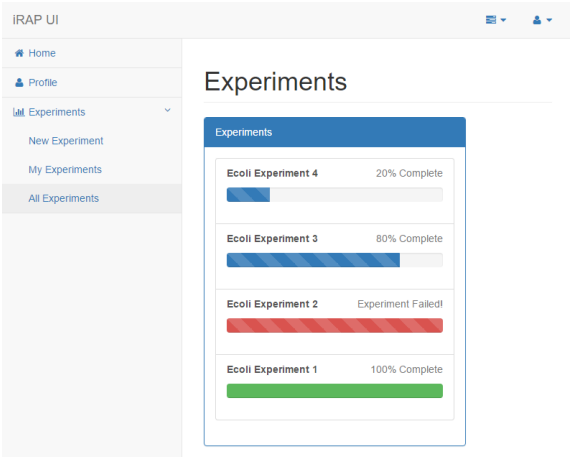

Figure 3.4: Experiments index of the UI module

Once a user selects an experiment, they will be shown that experiment's page, where all details and files are available. On this page a user can retrieve the execution logs for the experiment as well as the results in case any are available. On the Progress section, an estimate of completion is provided (or an error message if the experiment failed). Figure 3.5 shows both of these scenarios.

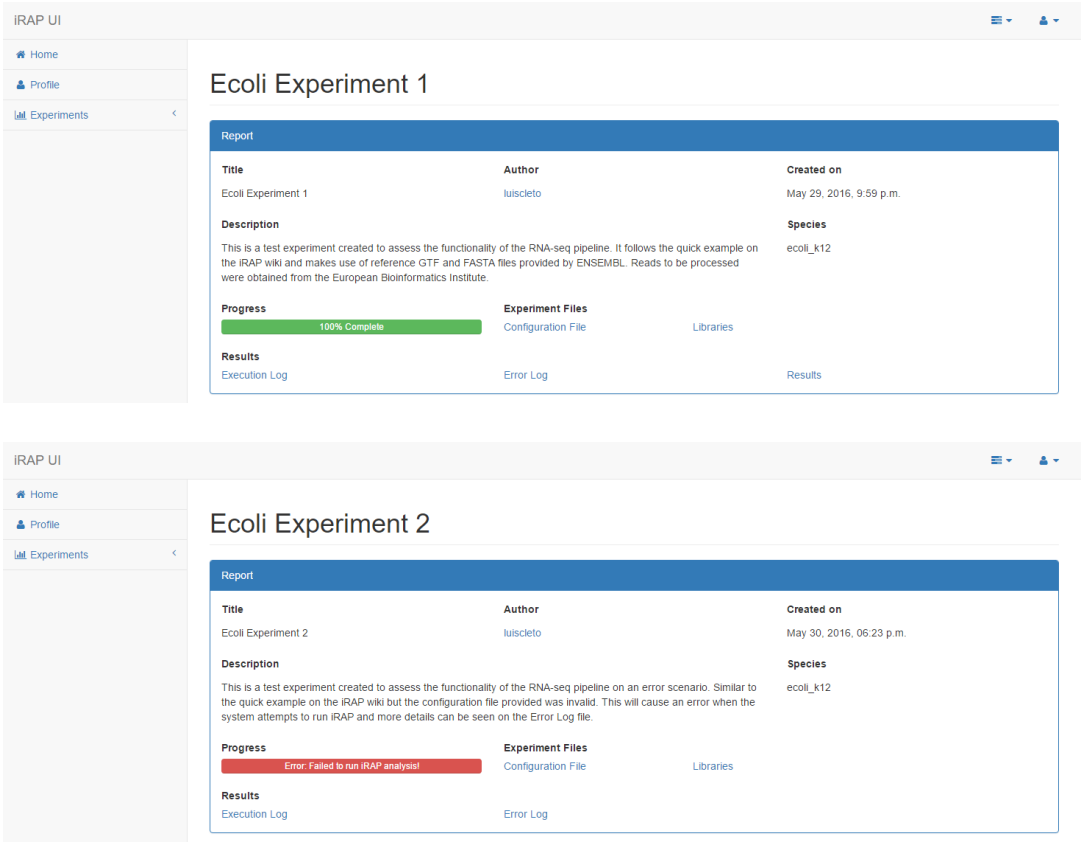

Figure 3.5: Experiment pages for a successful and a failed experiment

Finally, a user can also create a new experiment through the experiment creation form. As shown in Figure 3.6, this form requires a unique title for the experiment as well as a description. Additionally, a user must specify the reference files (GTF and annotation file) and provide both a configuration file for additional options and an archive containing the reads to be processed. Once an experiment is created, the front-end server will send a request to the Analysis module to start processing the requested experiment.

3.5 Analysis Server

Similarly to the front-end server, this module was developed using Python with Django. As with the previous ones, requirements should be installed and a *local_settings.py* file needs to be created to define the variables described in Table 3.2. This module should be deployed on the same physical machine as the iRAP system and the database specified in the standard settings should be the same as the one used by the front-end server.

Once the analysis server receives a request from the front-end server to start processing an experiment, it will begin retrieving that experiment's files and setup the experiment to allow running iRAP. It resorts to Python's *multiprocessing* module to maintain a pool of workers, along with a queue of experiments to execute, and uses the *subprocess* module to execute iRAP and analyse

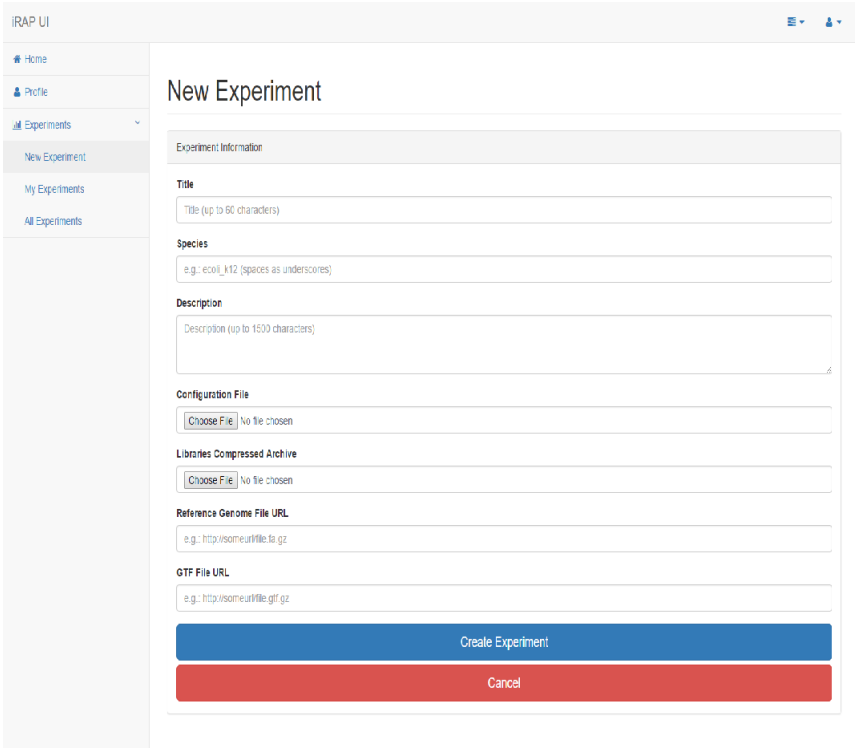

Figure 3.6: Experiment creation page

its output. As each step is completed, the status of the experiment is updated in the database. Once iRAP finishes executing, the results (if they were successfully generated) are compressed and stored in the database to allow access via the front-end server.

3.6 Case Studies

In order to validate the system's functionality as well as the iRAP installation, several experiments were created using reference data provided by the ENSEMBL project and raw reads from the European Bioinformatics Institute. In some of the experiments, errors were included in the input files to test the system's reliability.

3.6.1 Standard Experiment

Using the aforementioned input files together with a valid configuration file[6], the system was able to setup the experiment and run the full RNA-seq analysis successfully. At each step the experiment's status was updated so the progress could be tracked while it was ongoing. All the input and output files were stored in the database to allow retrieval by the end users.

[6] https://github.com/nunofonseca/irap/wiki/7-Quick-Example

3.6.2 Experiment With Errors

In addition to the previous experiment, a similar one was conducted but errors were introduced in the configuration file so that it would be invalid and iRAP would fail when executing the analysis. In this scenario, the system was able to detect the failure and provide a generic error message relative to the cause of the error (as seen in the second scenario of Figure 3.5). More details on errors during iRAP execution can be seen by analysing the provided output and error logs.

3.7 Chapter Conclusions

In this chapter we go over the architecture of the implemented system and detail the functions of each component/module. The main modules were implemented in Python and Django, using MongoDB for storing data with the GridFS system. The essentials were setup so that an expert can run RNA-seq experiments of any kind even though some options are currently only available via the text-based configuration file. Additionally, thanks to the decoupling of the interface, storage and core logic of the proposed system, it is simple to change the configuration of the computational resources at any time.

Chapter 4

RNA-seq Data Analysis

In order to provide additional meaningful insights into RNA-seq data, several types of analysis can be conducted which may be relevant for assessing the impact of cancer on the transcriptome. For this project, three main types of analysis were conducted: analysis of gene fusion events, alternative splicing and gene expression quantification (for mitochondrial genes). Multiple tools were considered or developed for this purpose as the remainder of this chapter will describe.

4.1 Gene Fusion Analysis

One of the types of analyses that were proposed was gene fusion analysis as a high amount of these events is commonly associated with cancer [MJFM15]. As there are several efficient tools available for this specific purpose already, we opted by configuring and using one such tool to conduct the required analysis. The EricScript (see Section 2.4) framework was chosen as it is considerably more lightweight than other tools and provides high quality results [KVQL16].

It is worth noting that, in order to use EricScript with the BAM samples provided for our research, these files have to first be sorted by query name and then converted back to FASTQ format. This can be achieved with the two commands shown in Listing 4.1.

```
1  samtools sort -n sample.bam sample.sorted.bam
2  bamToFastq -i sample.sorted.bam -fq sample.end1.fastq -fq2 sample.end2.fastq
```

Listing 4.1: Generating FASTQ input files for EricScript from a sample BAM file

4.2 Alternative Splicing Analysis

As was previously mentioned, one of the problems we aimed to solve is the development of a tool that allows measuring the events of alternative splicing present in given input samples. This

tool should be able to process vast amounts of data, as is usually the case for genetic analysis, within a reasonable amount of time and yield results that allow experts to visualize the differences in alternative splicing between each of the samples. In this chapter we present the *Junctions Comparer* tool, which was developed entirely by us, and detail its usage and methods.

4.2.1 Junctions Comparer

Once a set of reads has been aligned with a reference genome there are several types of analysis that can be performed on the alignments. One such analysis is quantifying the events of alternative splicing present in that sample as well as the types of such events. This is what *Junctions Comparer* aims to do by performing a differential analysis between a reference annotation genome and any number of aligned samples.

After the samples have been aligned, it is necessary to extract the splice junctions before using our tool to run the analysis. A junction is identified by its coordinates: chromosome, strand, start position and end position. This information can be extracted from a BAM file with a single command by combining SAMTools, AWK and BEDTools. This instruction can be seen in Listing 4.2.

```
samtools view -h sample_1.bam | awk '{if(($6 ~ /N/)|| $1 == "@SQ") print}' |
    samtools view -bS | bamToBed -bed12 | sort -k1,1V > sample_1.bed
```

Listing 4.2: Extracting splice sites from sample_1.bam

After pre-processing all the relevant samples, the resulting files in BED format can then be provided to *Junction Comparer* for quantifying and identifying the types of alternative splicing events in each sample. It is important to use the same reference genome that was used for the alignment process or the results may not be accurate.

4.2.2 Usage

Junctions Comparer was developed in Python and has no dependencies. It was designed to be as simple as possible to set up and use. The usage options are as seen in Listing 4.3.

```
usage: junctions_comparer.py [-h] [-nb | -q] [-n MIN_READS] [-t TEMP_DIR]
                             [-o OUT_DIR] [-r RESULTS_FILE]
                             [-rl RESULTS_DELIMITER] [-u UNKNOWN_ID]
                             [-gl GENE_DELIMITER] [-a {stranded,unstranded}]
                             gtf_file bed_file bed_file [bed_file ...]
```

Listing 4.3: Junctions Comparer Usage. Optional arguments are shown between square brackets and curly brackets contain the list of possible values for that argument

The required arguments for the script to execute are *gtf_file* and at least two *bed_files*. Every other argument is optional and were included primarily to give the user more customization options over the program's execution and output. The most relevant optional argument is the analysis type as it should be used in accordance to how the original experiment's libraries were prepared — using stranded protocols or not — as this needs to be correct for the results to be accurate. Table 4.1 describes each of the positional and optional arguments in detail.

Table 4.1: Junctions Comparer Arguments

Argument	**Description**
gtf_file	reference annotation file in GTF format
bed_file	junction annotation file in BED format
-h, –help	displays the usage information and exits
-nb, –no-bars	disables loading bars (use when redirecting output to file)
-q, –quiet	execute silently (disable all prints)
-n *MIN_READS*	minimum number of reads supporting a junction for it pass the filter (default: 3)
-t *TEMP_DIR*	directory where to keep temporary files (default: sj_tmp)
-o *OUT_DIR*	directory where to keep output files (default: sj_out)
-r *RESULTS_FILE*	name for final result file (default: results.csv)
-rl *RESULTS_DELIMITER*	delimiter for output CSV file (default: ,)
-u *UNKNOWN_ID*	ID given to reads in unknown genes (default: UNKNOWN)
-gl *GENE_DELIMITER*	delimiter for gene lists in output (default: |). NOTE: MUST NOT BE THE SAME AS *RESULTS_DELIMITER*
-a *ANALYSIS_TYPE*	stranded or unstranded analysis (default: unstranded)

4.2.3 Junction Identification

The aforementioned tool can identify canonical junctions as well as most of the classical types of alternative splicing events (see Section 2.1.3). At this stage, only mutually exclusive exons are not detected. Any such events will be detected as either canonical events (for consecutive exons) or standard exon skipping events (for non-consecutive exons).

In order to identify the type of a given junction, the tool will extract a list of known exons for the chromosome and strand that the junction belongs to. It will then analyse the start and end positions of the junction to determine if either is canonical or 3'/5' and also to see if the matching exons are consecutive or if an exon skipping event has taken place. By doing this it is also able to identify any combination of the classical types that were mentioned such as an alternative 3' acceptor site combined with an exon skipping event. Tables 4.2 and 4.3 show the possible classifications of a junction's start and end positions respectively. Additionally, a junction may also receive an additional exon skipping classification if the described scenarios do not occur between two consecutive exons.

Table 4.2: Junction Start Classification

Type	Description
Canonical	Occurs if the junction start matches an exon's end position
Alternative 5'	Occurs if the previous condition is not met but the junction start is contained within an exon
Novel	Occurs if none of the previous conditions are matched

Table 4.3: Junction End Classification

Type	Description
Canonical	Occurs if the junction end is immediately followed by an exon's start position
Alternative 3'	Occurs if the previous condition is not met but the position after a junction's end is contained within an exon
Novel	Occurs if none of the previous conditions are matched

As there are often overlapping exons in the reference genome (same position but different transcripts), precisely determining the type of junction being dealt with is often not trivial. The approach taken by the tool is that if any possible exon combination allows the junction to be considered canonical, that is the type that is assumed.

4.2.4 Output

Several output files are created by the program to summarize the results. They are in the *sj_out* folder unless the user specifies another output folder (see Section 4.2.2). A file named <*sample_name*>*.read_counts.txt* is created for each sample summarizing the number of junctions mapped to each gene (or unknown maps) and the total number of reads for that sample.

The main output files are *results.csv* and *results.filtered.csv* (again, unless a different file name was specified by the user). These files list the junctions found, the number of reads supporting that junction for each sample, the corresponding list of candidate gene IDs for both start and end positions of the junction and the type of junction. The *results.filtered.csv* only contains junctions that are supported by more reads than the *MIN_READS* parameter for at least one of the samples. Each junction is identified by a unique ID using one of the following schemes: <chromosome>_<start>_<end> for unstranded analysis or <chromosome>_<strand>_<start>_<end> for stranded analysis. An example of the output generated by this program is shown in Listing 4.4

```
1 id,sample_1,sample_2,sample_3,sample_4,gene_ids (start),gene_ids (end),type
2 01_154936407_154938114,1,3,0,4,ENSG00000160691,ENSG00000160691,C|ES|C
3 01_151542246_151546745,4,26,23,39,ENSG00000143367,ENSG00000143367,C|C
4 01_54675804_54678173,123,287,232,238,ENSG00000116221,ENSG00000116221,C|C
5 01_121307595_121310521,1,0,9,0,ENSG00000231752,ENSG00000231752,C|3'
6 01_27268309_27271880,3,3,1,6,ENSG00000090273,ENSG00000090273,C|ES|C
7 01_150621224_150621606,2,4,4,16,ENSG00000143457,ENSG00000143457,5'|C
```

```
8 01_7797630_7798018,2,3,2,2,ENSG00000171735,ENSG00000171735,C|C
9 01_52878296_52879550,76,142,201,195,ENSG00000134748|ENSG00000134744,ENSG00000134748
    |ENSG00000134744,C|C
```

Listing 4.4: Example output for an unstranded analysis. The first line contains the header of the CSV file (column names) while the following lines each identify a junction. The fields represent the junction's ID, the number of supporting reads per each sample, the IDs of the genes in the start and end positions and the type of junction.

4.2.5 Case Study

In order to evaluate the performance and quality of the results generated by *Junctions Comparer*, a simple case study was undertaken using both our tool and MATS (see Section 2.4) to perform a differential analysis on alternative splicing events between two samples as this is the only number of samples allowed for a MATS analysis. These samples were provided by IPATIMUP and both cases present a widely invasive Follicular Thyroid Carcinoma (wFTC). As the samples were prepared using unstranded libraries, the alternative splicing analysis is also, necessarily, unstranded, which might make the results of both tools less accurate. As the main purpose of the experiment was to compare results and performance, the analysis of the results themselves will be very superficial. In this study, JC was configured to filter out junctions with less than twelve supporting reads.

As Figure 4.1 shows, *Junctions Comparer* is significantly faster than MATS in processing the two samples, even if the pre-processing time of converting the BAM files to the required BED format is included. It is worth noting that the pre-processing was done in parallel for each sample so, for a significantly larger number of samples, this pre-processing time might increase significantly. It is likely that this large difference in execution time is due mainly to MATS being designed to perform the analysis on raw reads (FASTQ files) rather than BAM files.

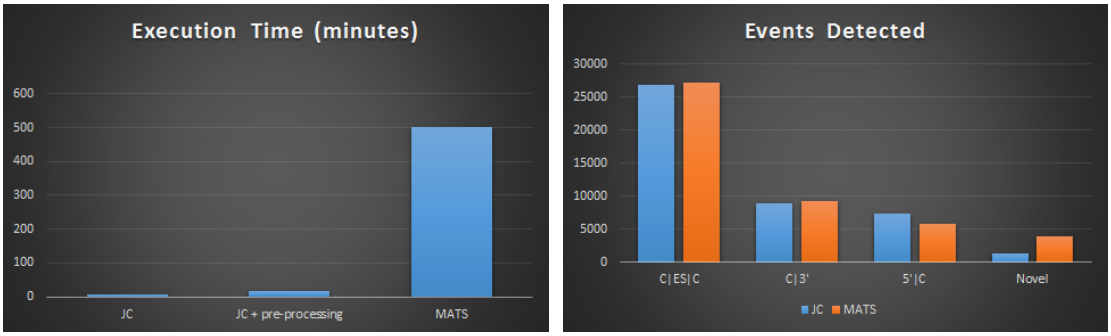

Figure 4.1: The graphic on the left shows the execution time for *Junctions Comparer* and MATS. The middle column includes the pre-processing time required to prepare the BAM samples for JC. The graphic on the right compares the results obtained by the two tools

As for the results generated by each tool, the most noticeable difference is in the number of novel junctions detected as MATS reports approximately three times as many novel junctions as JC. This is likely due to JC's optimistic approach in predicting splice events rather than conducting a more in-depth analysis using the read information to decide between ambiguous splice junctions.

4.3 Gene Expression Quantification

In addition to the *Junctions Comparer* tool, an extra script was developed to allow to easily quantify gene expression for each sample by using the total number of reads mapping to each gene as a proxy: the *Gene Expression Counter*. It accepts a subset of the options used for the previous script, the only difference being that it receives a custom format CSV file with the summarized reads rather than BED files. Listing 4.5 shows this program's usage.

```
usage: gene_expression_counter.py [-h] [-nb | -q] [-t TEMP_DIR] [-o OUT_DIR]
                                  [-rl RESULTS_DELIMITER] [-u UNKNOWN_ID]
                                  [-a {stranded,unstranded}]
                                  gtf_file reads_file reads_file [reads_file ...]
```

Listing 4.5: Gene Expression Counter usage. Optional arguments are shown between square brackets and curly brackets contain the list of possible values for that argument

The *reads_files* should contain one line per each read with the following values separated by tabs: <chromosome>, <read_start>, <read_end> and <strand>.

These files can be easily created from a BAM file by combining SAMTools and AWK through the command show in Listing 4.6.

```
samtools view sample.bam | awk '{printf $3 "\t" $4 "\t" $4+length($10)-1 "\t"} {if(
    and($2,16)>0){print "-"}else{print "+"}}'  > sample_reads.tsv
```

Listing 4.6: Extracting read information from sample.bam

4.3.1 Output

Gene Expression Counter creates a *<sample_name>_reads.gene_expression.tsv* for each sample with the following tab separated values: <gene_id>, <read_count> and <chromosome>. Additionally, the two last lines indicate the number of reads that don't map to any known gene and the total number of reads, respectively. These files are created in the *gec_out* directory unless another output directory is specified by the user. The results can then easily be filtered for specific chromosomes (such as mitochondrial chromosomes) using a text processing tool such as grep[1]. An example of such an output file is shown in Listing 4.7.

[1] https://www.gnu.org/software/grep/manual/grep.html

```
gene_id read_count  chr
ENSG00000099725 21  Y
ENSG00000183878 17  Y
ENSG00000241859 3   Y
UNKNOWN 3059        _
TOTAL   53844186    _
```

Listing 4.7: Example *Gene Expression Counter* output. The first line of the TSV file contains the header (column names) while every other line identifies a gene by its ID and shows its read count for this sample along with the gene's chromosome. The last two lines show the number of reads mapped to unknown locations and the total number of reads, respectively

4.4 Chapter Conclusions

In this chapter we go over each of the types of analyses proposed for our research. We detail the decisions taken for each analysis and the tools used or designed for that purpose. The tools that were created were *Junctions Comparer* and *Gene Expression Counter* and their usage and output were described in detail. The necessary pre-processing steps were referred in order to prepare BAM samples, like the ones provided by IPATIMUP, for each analysis.

Chapter 5

Case Study

In this chapter we will present the complete case study undertaken to assess the impact of cancer in several genomic features and events. The data used was provided by IPATIMUP and consists of four samples of thyroid carcinoma. The analyses performed and their methodology is explained in detail in Chapter 4.

The purpose of this case study was to research potential differences between distinct types of cancer. In order to accomplish this, four samples were collected from the São João Hospital Centre (Porto). Cases 1 and 2 have been categorized as a widely invasive Follicular Thyroid Carcinoma (wFTC) and cases 3 and 4 as a minimally invasive Follicular Thyroid Carcinoma (mFTC).

5.1 Samples characterization

Patient from case 1 was female, 82 years old, with a tumour measuring 2 cm of size and presented a predominant follicular growth pattern and oncocytic features. Patient from case 2 was male, 55 years old, with a tumour measuring 5 cm of size and presented a predominant solid/trabecular growth pattern. For the mFTC group, patient from case 3 was female, 56 years old, the tumour measuring 3.7 cm of size and with follicular growth pattern; patient from case 4 was female, 56 years old, the tumour measuring 4 cm of size and with follicular growth pattern.

5.2 Research Objectives

The aim of the study was to analyse the following features regarding the two sample groups:

- differentially expressed genes;
- genes most affected by alternative splicing events;
- most common gene fusion events;
- Compare results between versions of human genome assembly;

5.3 Analysis Protocol

The provided sample reads had already been aligned against reference genome GRCh37 with TopHat version 1.4.1. In order to allow a more extensive analysis and the use of different tools the original reads were extracted from the alignment files using the commands shown in Listing 5.1.

```
1    samtools sort -n sample.bam sample.sorted.bam
2    bamToFastq -i sample.sorted.bam -fq sample.end1.fastq -fq2 sample.end2.fastq
```

Listing 5.1: Extracting original reads from alignment files of pair-ended data

This also allowed us to redo the TopHat alignment using the most recent human genome annotation (GRCh38) so as to compare the results obtained with each genome version.

Using the data available, three major types of differential analyses were conducted: gene fusion, alternative splicing and gene expression. The methodology and processing steps used for generating the analysis results are described in detail in Chapter 4. In addition to these processing stages, we also developed an R[1] script to process the results (see Appendix A).

5.4 Genome version GRCh37 versus GRCh38

Even though IPATIMUP's data was assembled with reference genome version GRCh37, a newer version (GRCh38) has been released in late 2013. This new assembly is the first major revision of the human genome in more than four years[2]. The main differences between the two versions are:

- **Alternate sequences**: Several human chromosomal regions exhibit sufficient variability to prevent adequate representation by a single sequence. To address this, the GRCh38 assembly provides alternate sequence for selected variant regions through the inclusion of alternate loci[3] scaffolds.
- **Centromere[4] representation**: The previous method of representing centromeric regions (large megabase-sized gaps) has been replaced by sequences from centromere models. The models provide the approximate repeat number and order for each centromere and are useful for read mapping and variation studies.
- **Mitochondrial genome**: The mitochondrial reference sequence included in the GRCh38 assembly is the Revised Cambridge Reference Sequence (rCRS) with RefSeq accession number NC_012920.1. This differs from the previous sequence (RefSeq accession number NC_001907) which was not updated when the GRCh37 assembly later transitioned to the new version.

[1] https://www.r-project.org/

[2] https://groups.google.com/a/soe.ucsc.edu/forum/#!topic/genome-announce/52Kv41YBXNY

[3] Specific location on a chromosome

[4] the point on a chromosome by which it is attached to a spindle fibre during cell division

- **Sequence updates**: Several erroneous bases and misassembled regions in GRCh37 have been corrected in the GRCh38 assembly, and more than 100 gaps have been filled or reduced.

- **Analysis set**: The GRCh38 assembly offers an "analysis set" that was created to accommodate next generation sequencing read alignment pipelines. Several GRCh38 regions have been eliminated from this set to improve read mapping.

5.5 Results

5.5.1 Gene Expression Analysis

As a starting point for our analysis, we first took into consideration the results obtained by the research of Celestino et al.[5] as they did an analysis of the same samples used for this study using the GRCh37 reference annotation. According to the authors, there were seventeen genes differentially expressed between mFTC and wFTC. Of those seventeen, genes C1QL1, LCN2, CRABP1 and CILP were also differentially expressed when compared to normal thyroid samples.

We have verified these results using genome version GRCh37. However, using the data generated with the updated annotation (GRCh38), the results differ considerably as all of the genes in the detected cases are no longer considered expressed. These values can be seen in Table 5.1.

Table 5.1: Gene Expression Results for both sample groups using GRCh37 and GRCh38. The values for each column represent the average of the normalized fraction of reads that mapped to that gene in each sample

gene symbol	wFTC (37)	mFTC (37)	wFTC (38)	mFTC (38)
C1QL1	0,000461013	6,91069E-06	0	0
LCN2	0,003166684	1,77554E-05	0	0
CRABP1	1,17362E-05	0,004140188	0	0
CILP	1,49025E-05	0,001517215	0	0

We have also found sixteen differentially expressed genes using only the newest genome annotation. The results we obtained can be seen in Table 5.2. Additionally, a brief summary of each gene detected is available in Appendix B.1. It is worth noting that none of these are mitochondrial genes as no significative difference was found between those in each sample group, although these genes had considerably higher expression values in sample 1 than each of the other 3 samples.

[5]Ricardo Celestino, Torfin Nome, Ana Pestana, Andreas M. Hoff, Pedro A. Gonçalves, Catarina Eloy, Eva Sigstad, Ragnhild A. Lothe, Trine Bjøro, Manuel Sobrinho-Simões, Rofl I. Skotheim, Paula Soares, *CRAPBP1, C1QL1 and LCN2 are biomarkers of thyroid cancer and predictors of extrathyroidal extension*, submitted for publication

Table 5.2: Gene Expression results for both sample groups using GRCh38. The values for the second and third column represent the average of the normalized fraction of reads that mapped to that gene in each sample. The last column represents the largest difference factor calculated as *MAX(wFTC/mFTC, mFTC/wFTC)*

gene symbol	**wFTC**	**mFTC**	**Factor**
ATP1A3	0,002741105	8,48341E-06	323,1134112
NXPH4	0,002550934	1,68349E-05	151,526695
EDN1	1,2982E-05	0,001626718	125,3056145
ASXL3	1,78247E-05	0,001909248	107,1124767
RP11-146G7.2	5,32011E-06	0,000557117	104,7190109
RP11-742D12.2	0,000385299	4,30768E-06	89,44472411
MAG	2,50092E-06	0,000200977	80,36116137
NLRP2	5,03366E-05	0,003740218	74,30410934
KIF5A	1,76656E-05	0,00127871	72,384344
RP11-109I13.2	0,000217341	4,30768E-06	50,45419146
NPTX2	0,000533217	1,06373E-05	50,12731118
PHEX-AS1	1,25046E-06	6,07247E-05	48,56196168
AMD1P4	5,00183E-06	0,000241634	48,30902915
ZCCHC12	0,000631526	2,12085E-05	29,77698653
COL17A1	0,000300223	0,008407357	28,00372591
HCAR1	3,74568E-05	0,0010292	27,47695531

5.5.2 Alternative Splicing Analysis

In order to provide a differential analysis of alternative splicing events between the two sample groups we used an R script (see Appendix A) to process the large output of *Junctions Comparer* and extend the results with some useful visualizations. The thirty most notable results for GRCh38 can be seen in Figure 5.1.

Appendix B.2 lists the differentiated cases when compared between each sample group, and the respective factors, while Appendix B.3 provides a brief description of the functions identified for each of the genes that show significative differential alternative splicing.

It is worth noting that while many of the identified genes also stand out when GRCh37 is used (the top 30 results share 20 common genes in each experiment), there are significant differences as only one gene is ranked in the same position in both experiments — the most differentially spliced gene in both samples. As with the previous analysis, this shows the importance of using the most updated assembly of the human genome for experiments.

5.5.3 Gene Fusion Analysis

For the gene fusion analysis we again used the work of Celestino et al. as a starting point as they had identified and experimentally validated several fusion pairs.

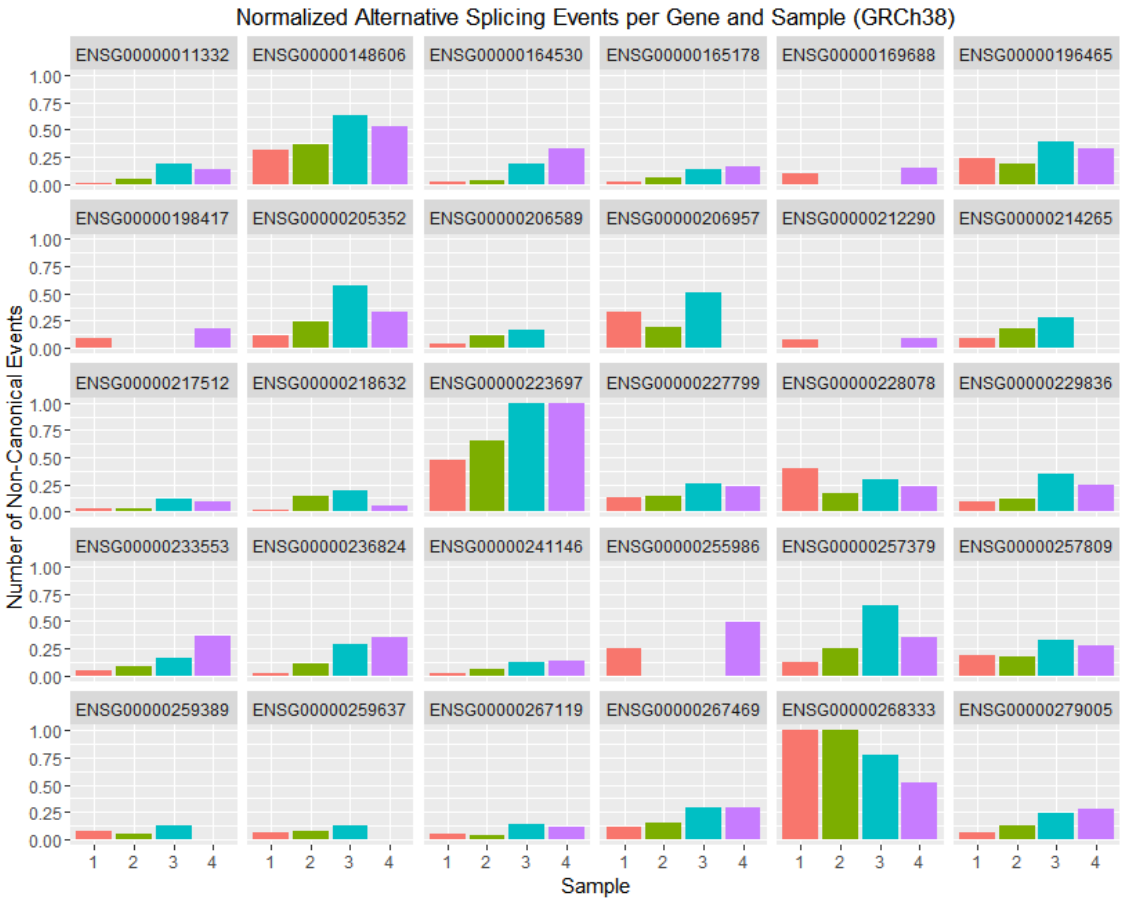

Figure 5.1: Most significant differentially spliced genes between wFTC and mFTC sample groups. Each graphic shows the normalized amount of alternative splicing events per sample for each gene. These results were obtained using the GRCh38 genome assembly

We used EricScript and ran the analysis using both GRCh37 and GRCh38 and, while the updated version yielded results closer to what was expected, most of the cases identified and experimentally validated in the previous work were not detected. We were only able to confirm 2 of the aforementioned fusion events and a third one in only 2 of the 3 samples it was detected in. The comparison between the experiments is shown in Table 5.3.

This leads us to conclude that EricScript does not yield accurate results when handling real data in comparison to its outstanding performance with synthetic data. This observation is consistent with the study performed by Liu et al. [LTD+16] which concludes that EricScript outperforms most tools using synthetic data but performs poorly in real data sets.

5.6 Chapter Conclusions

This chapter introduces the case study performed to validate the methodology proposed in Chapter 4 and also to provide new insights on the distinguishing genomic features of the two types of

Table 5.3: Gene Fusion events detected in the original study and by EricScript using both GRCh37 and GRCh38. The gene symbols used to identify the fusion genes are the ones used in the older annotation for compatibility with the previous paper

Fusion Gene (old symbols)	Chr1	Chr2	Samples (Original)	Samples (GRCh37)	Samples (GRCh38)
DISP1-SUPT20H	1	13	2, 4	-	-
EML2-C16orf46	19	16	1	-	-
FBXO25-BET1L	8	11	4	-	4
FBXO25-RP11-261C10.34	8	1	2	-	-
GSN-KIAA0586	9	14	2, 4	-	-
HIBCH-ERI2	2	16	2, 4	-	-
NUBPL-PPP1R3F	14	X	3	-	-
PHKA2-SYTL3	X	6	4	-	-
PPP1R3F-NUBPL	X	14	1, 2, 3	-	1, 3
RBM27-FCGBP	5	19	2	-	-
SAV1-GYPE	14	4	3	3	3
SCRN3-SCFD1	2	14	4	-	-
SCRN3-RABGAP1L	2	1	2	-	-
SLC22A20-PPARD	11	6	3	-	-
C1orf196-KAZN	1	1	1	-	-
KIAA1267-ARL17A5	17	17	1	-	-
KIAA1267-ARL17B5	17	17	4	-	-
LOC728613-SDHA	5	5	4	-	-
MIR4435-1HG-ANAPC1	2	2	3, 4	-	-
RP11-141M1.4-STARD13	13	13	2, 3, 4	-	-
RP11-634B7.4-TRIM58	1	1	1	-	-

thyroid cancer that were studied.

Four thyroid cancer samples were collected and divided into two distinct groups based on the type of cancer. We showed that the differences in results are significative when using GRCh38 instead of its older counterpart GRCh37. We identified the genes with highest differential expression between the two samples and also those with the highest differential alternative splicing events. We also verified some gene fusion events but concluded that the EricScript tool does not yield accurate results when handling real data rather than synthetic data.

Chapter 6

Conclusions and Future Work

In this chapter we will review the general objectives of the dissertation and the goals that we achieved as well as the contributions that were made. Additionally, we also present suggestions for future work.

6.1 Conclusions

The analysis of human transcriptomes is a computationally intensive task which requires not only powerful computational resources but also the combination of multiple complex tools to yield the desired results. As such, it can be difficult for geneticists and molecular biologists to run such analyses themselves, often requiring the assistance of a third-party such as an IT expert or even a company. To this end we developed the eRAP system, a prototype system built on top of the iRAP RNA-seq pipeline which allows the user to run the analysis via a web interface without having to worry about the configuration of the underlying system.

In addition to this system, we also developed the *Junctions Comparer* and *Gene Expression Counter* tools for performing differential alternative splicing and gene expression analysis, respectively. They are simple approaches to each scenario, capable of yielding consistent results within a relatively short time frame when compared to existing tools with similar purpose.

Using the developed methodology, we have analysed real data provided by IPATIMUP's researchers to address several research questions. We have identified the most differentially expressed genes as well as the ones with the most different alternative splicing events between the two sample groups that were considered. We were also able to identify some of the gene fusion events that had been experimentally verified but concluded that the EricScript tool used for this analysis does not yield accurate results when handling real data. Finally, we also showed that using an outdated version of the human genome can yield significantly different results and compromise the conclusions that can be drawn from them.

6.2 Future Work

The work done for this dissertation can be improved in several aspects. The eRAP system is currently only a prototype and can be extended to include more configuration options in the interface until the need to upload an additional configuration file is completely eliminated. Additionally, the tools we developed during this research can also be included in this system (or the underlying iRAP system) in order to allow more types of analyses. It would also be beneficial to separate experiment creation and large file uploads on this system to avoid having to re-upload the files if there is an error in the form (this implies allowing users to manage their files).

The *Junctions Comparer* and *Gene Expression Counter* tools can also be improved by including statistical analysis of the results generated in order to avoid having to develop additional R scripts or the use of spreadsheet software to summarize the results, as was done in this work. It would also be useful to calculate and include confidence metrics for each event rather than relying on the absolute number of reads that support each event.

More types of analyses can also be performed using the tools that were developed, for example by taking into account the types of the alternative splicing events which are also included in the *Junctions Comparer* output. Finally, the gene fusion analysis should follow a different methodology using a tool capable of providing more accurate results when handling real data such as SOAPfuse, FusionCatcher, JAFFA or PRADA.

References

[A+10] Simon Andrews et al. Fastqc: A quality control tool for high throughput sequence data. *Reference Source*, 2010.

[AKW78] Alfred V. Aho, Brian W. Kernighan, and Peter J. Weinberger. Awk – a pattern scanning and processing language (second edition), 1978.

[APH14] Simon Anders, Paul Theodor Pyl, and Wolfgang Huber. Htseq—a python framework to work with high-throughput sequencing data. *Bioinformatics*, page btu638, 2014.

[ASK+01] Shinichiro Ando, Nicholas J Sarlis, Jay Krishnan, Xu Feng, Samuel Refetoff, Michael Q Zhang, Edward H Oldfield, and Paul M Yen. Aberrant alternative splicing of thyroid hormone receptor in a tsh-secreting pituitary tumor is a mechanism for hormone resistance. *Molecular Endocrinology*, 15(9):1529–1538, 2001.

[BPM+12] Matteo Benelli, Chiara Pescucci, Giuseppina Marseglia, Marco Severgnini, Francesca Torricelli, and Alberto Magi. Discovering chimeric transcripts in paired-end RNA-seq data by using EricScript. *Bioinformatics*, 28(24):3232–3239, December 2012.

[Bro15] Ryan Brown. Django vs flask vs pyramid: Choosing a python web framework. 2015.

[CFG+10] Peter JA Cock, Christopher J Fields, Naohisa Goto, Michael L Heuer, and Peter M Rice. The sanger fastq file format for sequences with quality scores, and the solexa/illumina fastq variants. *Nucleic acids research*, 38(6):1767–1771, 2010.

[ED] Alicia Ebert and Gene Delay. Mitochondrial DNA Analysis. Available at http://www.uvm.edu/~biology/Classes/296D/10_Mitochondria.pdf, accessed on June 2016.

[FCK02] Lilian TC Franca, Emanuel Carrilho, and Tarso BL Kist. A review of dna sequencing techniques. *Quarterly reviews of biophysics*, 35(02):169–200, 2002.

[FG08] James D. Fackenthal and Lucy A. Godley. Aberrant rna splicing and its functional consequences in cancer cells. *Disease Models and Mechanisms*, 1(1):37–42, 2008.

[FPMB14] Nuno A Fonseca, Robert Petryszak, John Marioni, and Alvis Brazma. irap-an integrated rna-seq analysis pipeline. *bioRxiv*, page 005991, 2014.

[Gena] Generic Model Organism Database. GFF. Available at http://gmod.org/wiki/GFF, accessed on June 2016.

[GENb] GENIE. Gene Expression and Regulation. Available at http://www2.le.ac.uk/departments/genetics/vgec/schoolscolleges/topics/geneexpression-regulation, accessed on June 2016.

[gro] UCSC Genome Bioinformatics group. BED Format Specification. Available at http://genome.ucsc.edu/FAQ/FAQformat#format1, accessed on June 2016.

[GYLL13] Marieta Gencheva, Lixin Yang, Gong-Biao Lin, and Ren-Jang Lin. Detection of alternatively spliced or processed rnas in cancer using oligonucleotide microarray. In *RNA and Cancer*, pages 25–40. Springer, 2013.

[HKA] Galit Lev-Maor Hadas Keren and Gil Ast. Different types of alternative splicing. Available at http://www.nature.com/nrg/journal/v11/n5/box/nrg2776_BX1.html, accessed on June 2016.

[Ins] National Human Genome Research Institute. The ENCODE Project: ENCyclopedia Of DNA Elements. Available at http://www.genome.gov/encode/, accessed on June 2016.

[KVQL16] Shailesh Kumar, Angie Duy Vo, Fujun Qin, and Hui Li. Comparative assessment of methods for the fusion transcripts detection from rna-seq data. *Scientific reports*, 6, 2016.

[LHW+09] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, et al. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[LM13] Yishan Li and Sathiamoorthy Manoharan. A performance comparison of sql and nosql databases. In *Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on*, pages 15–19. IEEE, 2013.

[LS12] Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357–359, 2012.

[LTD+16] Silvia Liu, Wei-Hsiang Tsai, Ying Ding, Rui Chen, Zhou Fang, Zhiguang Huo, SungHwan Kim, Tianzhou Ma, Ting-Yu Chang, Nolan Michael Priedigkeit, et al. Comprehensive evaluation of fusion transcript detection algorithms and a meta-caller to combine top performing methods in paired-end rna-seq data. *Nucleic acids research*, 44(5):e47–e47, 2016.

[Man14] Ananya Mandal. What is Molecular Biology? Available at http://www.news-medical.net/life-sciences/What-is-Molecular-Biology.aspx, accessed on June 2016, October 2014.

[Mar08] Elaine R. Mardis. The impact of next-generation sequencing technology on genetics. *Trends in Genetics*, 24(3):133 – 141, 2008.

[MHZ+11] Andrew McPherson, Fereydoun Hormozdiari, Abdalnasser Zayed, Ryan Giuliany, Gavin Ha, Mark GF Sun, Malachi Griffith, Alireza Heravi Moussavi, Janine Senz, Nataliya Melnyk, et al. defuse: an algorithm for gene fusion discovery in tumor rna-seq data. *PLoS Comput Biol*, 7(5):e1001138, 2011.

[MJFM15] Fredrik Mertens, Bertil Johansson, Thoas Fioretos, and Felix Mitelman. The emerging complexity of gene fusions in cancer. *Nature Reviews Cancer*, 15(6):371–381, 2015.

[MJM07] Felix Mitelman, Bertil Johansson, and Fredrik Mertens. The impact of translocations and gene fusions on cancer causation. *Nature Reviews Cancer*, 7(4):233–245, 2007.

[NCB] NCBI. Web BLAST page options. Available at http://blast.ncbi.nlm.nih.gov/blastcgihelp.shtml, accessed on June 2016.

[QH10] Aaron R Quinlan and Ira M Hall. Bedtools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–842, 2010.

[Rei] Reid J. Robinson. How big is the human genome? Available at https://medium.com/precision-medicine/how-big-is-the-human-genome-e90caa3409b0, accessed on June 2016.

[Sch07] Stephan C Schuster. Next-generation sequencing transforms today's biology. *Nature*, 200(8):16–18, 2007.

[SPL+14] Shihao Shen, Juw W. Park, Zhi-xiang Lu, Lan Lin, Michael D. Henry, Ying N. Wu, Qing Zhou, and Yi Xing. rMATS: Robust and flexible detection of differential alternative splicing from replicate RNA-Seq data. *Proceedings of the National Academy of Sciences*, 111(51):E5593–E5601, December 2014.

[TPS09] Cole Trapnell, Lior Pachter, and Steven L Salzberg. Tophat: discovering splice junctions with rna-seq. *Bioinformatics*, 25(9):1105–1111, 2009.

[TRG+12] Cole Trapnell, Adam Roberts, Loyal Goff, Geo Pertea, Daehwan Kim, David R Kelley, Harold Pimentel, Steven L Salzberg, John L Rinn, and Lior Pachter. Differential gene and transcript expression analysis of rna-seq experiments with tophat and cufflinks. *Nature protocols*, 7(3):562–578, 2012.

[Ven04] Julian P Venables. Aberrant and alternative splicing in cancer. *Cancer research*, 64(21):7647–7654, 2004.

[WGS09] Zhong Wang, Mark Gerstein, and Michael Snyder. Rna-seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, 2009.

[Wol13] Jochen BW Wolf. Principles of transcriptome analysis and gene expression quantification: an rna-seq tutorial. *Molecular ecology resources*, 13(4):559–572, 2013.

REFERENCES

Appendix A

R Analysis

This document contains the R script developed to analyse the data generated by *Junctions Comparer* for the case study described in Chapter 5. The script's dependencies are also listed.

A.1 Dependencies

In order to run the script in the next section, the following R dependencies should be installed:

- *reshape2*[1] - an R package to flexibly rearrange, reshape and aggregate data
- *ggplot2*[2] - an implementation of the grammar of graphics in R

Additionally, the absolute paths present in the script should be changed to match those in the system being used for the analysis.

A.2 Script Code

```
1  library(reshape2)
2  library(ggplot2)
3
4  gene_ids = readLines("D:\\FEUP\\5o_ano\\2o_semestre\\DISS\\AltSplicing\\geneids_new
       .txt")
5  alt_data = read.csv("D:\\FEUP\\5o_ano\\2o_semestre\\DISS\\junctions_comparer\\
       sj_out\\results.filtered.csv", header = TRUE, sep = ",")
6  samp1 = subset(alt_data, sample_1 > 50)
7  samp2 = subset(alt_data, sample_2 > 50)
8  samp3 = subset(alt_data, sample_3 > 50)
9  samp4 = subset(alt_data, sample_4 > 50)
10 samp1_alt = subset(samp1, type != "C|C")
```

[1] https://cran.r-project.org/web/packages/reshape2/reshape2.pdf
[2] https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf

```
samp2_alt = subset(samp2, type != "C|C")
samp3_alt = subset(samp3, type != "C|C")
samp4_alt = subset(samp4, type != "C|C")

sample1_events = vector(mode="integer",length(gene_ids))
i = 0
for (gen_id in gene_ids) {
    sample1_events[i] <- sum(grepl(gen_id, samp1_alt$gene_ids..start.) | grepl(
        gen_id, samp1_alt$gene_ids..end.))
    i <- i + 1
}

sample2_events = vector(mode="integer",length(gene_ids))
i = 0
for (gen_id in gene_ids) {
    sample2_events[i] <- sum(grepl(gen_id, samp2_alt$gene_ids..start.) | grepl(
        gen_id, samp2_alt$gene_ids..end.))
    i <- i + 1
}

sample3_events = vector(mode="integer",length(gene_ids))
i = 0
for (gen_id in gene_ids) {
    sample3_events[i] <- sum(grepl(gen_id, samp3_alt$gene_ids..start.) | grepl(
        gen_id, samp3_alt$gene_ids..end.))
    i <- i + 1
}

sample4_events = vector(mode="integer",length(gene_ids))
i = 0
for (gen_id in gene_ids) {
    sample4_events[i] <- sum(grepl(gen_id, samp4_alt$gene_ids..start.) | grepl(
        gen_id, samp4_alt$gene_ids..end.))
    i <- i + 1
}


sample1_events = (sample1_events-min(sample1_events))/(max(sample1_events)-min(
    sample1_events))
sample2_events = (sample2_events-min(sample2_events))/(max(sample2_events)-min(
    sample2_events))
sample3_events = (sample3_events-min(sample3_events))/(max(sample3_events)-min(
    sample3_events))
sample4_events = (sample4_events-min(sample4_events))/(max(sample4_events)-min(
    sample4_events))

gen_alt_splice = data.frame(gene_ids, sample1_events, sample2_events,
    sample3_events, sample4_events)

```

```
gen_alt_splice[, "max12"] <- apply(gen_alt_splice[, 2:3], 1, max)
gen_alt_splice[, "max34"] <- apply(gen_alt_splice[, 4:5], 1, max)
gen_alt_splice[, "min12"] <- apply(gen_alt_splice[, 2:3], 1, min)
gen_alt_splice[, "min34"] <- apply(gen_alt_splice[, 4:5], 1, min)
gen_alt_splice[, "diff1"] <- abs(gen_alt_splice[, 9] - gen_alt_splice[, 6])
gen_alt_splice[, "diff2"] <- abs(gen_alt_splice[, 8] - gen_alt_splice[, 7])
gen_alt_splice[, "diff"] <- apply(gen_alt_splice[, 10:11], 1, min)
largest_diffs = gen_alt_splice[order(gen_alt_splice[,12],decreasing=T), ]

melted <- melt(largest_diffs[1:20,1:5], "gene_ids")
melted$cat <- ''
melted[melted$variable == 'sample4_events',]$cat <- "4"
melted[melted$variable == 'sample3_events',]$cat <- "3"
melted[melted$variable == 'sample2_events',]$cat <- "2"
melted[melted$variable == 'sample1_events',]$cat <- "1"

ggplot(melted, aes(x = cat, y = value, fill = variable)) +
    geom_bar(stat = 'identity',position = 'stack') +
    facet_wrap(~ gene_ids) +
    labs(title="Normalized Alternative Splicing Events per Gene and Sample", y="
        Number of Non-Canonical Events", x="Sample", fill="Samples")
```

Appendix B

Experiment Results

This document contains additional experiment data such as gene descriptions for the genes identified in each analysis performed for the case study.

B.1 Gene Expression Results' Descriptions

gene symbol	Gene Info
ATP1A3	encodes protein belonging to the family of P-type cation transport ATPases, and to the subfamily of Na+/K+ -ATPases
NXPH4	encodes protein which may be signaling molecules that resemble neuropeptides and that act by binding to alpha-neurexins and possibly other receptors
EDN1	encodes a preproprotein that is proteolytically processed to generate a secreted peptide that belongs to the endothelin/sarafotoxin family. This peptide is a potent vasoconstrictor and its cognate receptors are therapeutic targets in the treatment of pulmonary arterial hypertension. Aberrant expression of this gene may promote tumorigenesis
ASXL3	codes PcG proteins which act by forming multiprotein complexes, which are required to maintain the transcriptionally repressive state of homeotic genes throughout development
RP11-146G7.2	known antisense
RP11-742D12.2	known processed transcript
MAG	encodes protein which is a type I membrane protein and member of the immunoglobulin superfamily. It is thought to be involved in the process of myelination. It is a lectin that binds to sialylated glycoconjugates and mediates certain myelin-neuron cell-cell interactions

NLRP2	NALP proteins, such as NALP2, are characterized by an N-terminal pyrin domain (PYD) and are involved in the activation of caspase-1 by Toll-like receptors. They may also be involved in protein complexes that activate proinflammatory caspases
KIF5A	encodes a member of the kinesin family of proteins. Members of this family are part of a multisubunit complex that functions as a microtubule motor in intracellular organelle transport
RP11-109I13.2	known processed transcript
NPTX2	encodes a member of the family of neuronal petraxins, synaptic proteins that are related to C-reactive protein. This protein is involved in excitatory synapse formation
PHEX-AS1	PHEX antisense RNA 1 (non-protein coding)
AMD1P4	adenosylmethionine decarboxylase 1 pseudogene 4
ZCCHC12	encodes a downstream effector of bone morphogenetic protein (BMP) signalling. This protein contains a zinc finger domain and functions as a transcriptional coactivator. Variation in this gene may be associated with X-linked mental retardation
COL17A1	encodes the alpha chain of type XVII collagen
HCAR1	G protein-coupled receptors (GPCRs, or GPRs), such as GPR81, contain 7 transmembrane domains and transduce extracellular signals through heterotrimeric G proteins

Table B.1: Short description of genes that are differentially expressed between samples mFTC and wFTC

B.2 Alternative Splicing Results

gene symbol	**Gene ID**	**wFTC**	**mFTC**	**Factor**
AF230666.2	ENSG00000223697	0,55985552	1	1,786175119
MT1JP	ENSG00000255986	0,12352941	0,24358974	1,971916971
RP4-806M20.3	ENSG00000268333	1	0,65014446	1,538119697
BCYRN1	ENSG00000236824	0,06439628	0,31392199	4,874846653
POLR3A	ENSG00000148606	0,33864809	0,57972192	1,711871223
PI16	ENSG00000164530	0,02930857	0,25180571	8,591538584
AC005944.2	ENSG00000267469	0,1377709	0,29532322	2,14358199
XXbac-BPG248L24.10	ENSG00000229836	0,10407637	0,29785121	2,861852407
RP5-1186P10.1	ENSG00000279005	0,09520124	0,26074395	2,738871363
MT1F	ENSG00000198417	0,04705882	0,08974359	1,907051431
RP11-793H13.8	ENSG00000257379	0,18601651	0,49702059	2,671916541

RPL7P28	ENSG00000218632	0,07605779	0,12423257	1,633397052
DPF1	ENSG00000011332	0,03219814	0,16206212	5,033275835
PRR13	ENSG00000205352	0,18163055	0,45539906	2,507282283
MYL6B	ENSG00000196465	0,20975232	0,35743951	1,704102772
AC012358.4	ENSG00000227799	0,13926729	0,24214518	1,738708206
RP11-603J24.14	ENSG00000257809	0,18183695	0,29658722	1,631061344
NCF1C	ENSG00000165178	0,04246646	0,15375587	3,620642502
AC108462.1	ENSG00000233553	0,0630031	0,25695197	4,078402015
RPL7P41	ENSG00000241146	0,0380805	0,1338931	3,516054148
RPL10P15	ENSG00000267119	0,04107327	0,12811484	3,119177996
RP11-477E3.2	ENSG00000217512	0,0249226	0,10120982	4,060965549

Table B.2: Alternative Splicing analysis results for both sample groups using GRCh38. The values for the third and fourth column represent the average of the normalized number of alternative splicing events that took place within that gene in each sample. The last column represents the largest difference factor calculated as *MAX(wFTC/mFTC, mFTC/wFTC)*

B.3 Alternative Splicing Results' Descriptions

gene symbol	**Gene ID**	**Gene Info**
AF230666.2	ENSG00000223697	associated with TG (Thyroglobulin) and PHF20L1 genes. Thyroglobulin (Tg) is a glycoprotein homodimer produced predominantly by the thryroid gland
MT1JP	ENSG00000255986	metallothionein 1J, pseudogene. MTs have the capacity to bind both physiological (such as zinc, copper, selenium) and xenobiotic (such as cadmium, mercury, silver, arsenic) heavy metals through the thiol group of its cysteine residues
RP4-806M20.3	ENSG00000268333	lincRNA
BCYRN1	ENSG00000236824	encodes a neural small non-messenger RNA, is a member of the family of interspersed repetitive DNA, and its product represents an example of a primate tissue-specific RNA polymerase III transcript. It is believed that this gene was retropositionally generated and recruited into a function regulating dendritic protein biosynthesis

POLR3A	ENSG00000148606	encodes the catalytic component of RNA polymerase III, which synthesizes small RNAs. The encoded protein also acts as a sensor to detect foreign DNA and trigger an innate immune response
PI16	ENSG00000164530	encodes a putative serine protease inhibitor
AC005944.2	ENSG00000267469	known antisense
XXbac-BPG248L24.10	ENSG00000229836	known unprocessed pseudogene
RP5-1186P10.1	ENSG00000279005	known TEC (to be experimentally confirmed)
MT1F	ENSG00000198417	encodes a protein of the metallothionein family. Metallothioneins have a high content of cysteine residues that bind various heavy metals; these proteins are transcriptionally regulated by both heavy metals and glucocorticoids
RP11-793H13.8	ENSG00000257379	known processed transcript
RPL7P28	ENSG00000218632	ribosomal protein L7 pseudogene 28
DPF1	ENSG00000011332	the protein it codes may have an important role in developing neurons by participating in regulation of cell survival, possibly as a neurospecific transcription factor. Belongs to the neuron-specific chromatin remodeling complex (nBAF complex)
PRR13	ENSG00000205352	encodes a protein which negatively regulates TSP1 expression at the level of transcription. This down-regulation was shown to reduce taxane-induced apoptosis
MYL6B	ENSG00000196465	myosin is a hexameric ATPase cellular motor protein. It is composed of two heavy chains, two non-phosphorylatable alkali light chains, and two phosphorylatable regulatory light chains. This gene encodes a myosin alkali light chain expressed in both slow-twitch skeletal muscle and in nonmuscle tissue
AC012358.4	ENSG00000227799	known processed pseudogene (possibly associated with CLHC1)
RP11-603J24.14	ENSG00000257809	known antisense

NCF1C	ENSG00000165178	encodes the 47 kDa cytosolic subunit of neutrophil NADPH oxidase, which produces superoxide anion. The NCF1 gene is located in close proximity to two highly similar, multi-exon pseudogenes. Recombination events between the pseudogenes and the functional NCF1 gene can inactivate the NCF1 gene and result in chronic granulomatous disease
AC108462.1	ENSG00000233553	known antisense
RPL7P41	ENSG00000241146	ribosomal protein L7 pseudogene 41
RPL10P15	ENSG00000267119	RPL10P15 ribosomal protein L10 pseudogene
RP11-477E3.2	ENSG00000217512	known processed pseudogene

Table B.3: Short description of genes that show differential alternative splicing between samples mFTC and wFTC