8-2012

A systems-based approach for detecting molecular interactions across tissues.

Fahim Mohammad
University of Louisville

A SYSTEMS–BASED APPROACH FOR DETECTING MOLECULAR INTERACTIONS ACROSS TISSUES

By

Fahim Mohammad
B. Tech. (CSE), AMU Aligarh, India, 2001
M. Tech. (IT), IPU Delhi, India, 2007

A Dissertation submitted to the
J. B. Speed School of Engineering
in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, KY
USA - 40209

August 2012

A SYSTEMS-BASED APPROACH FOR DETECTING MOLECULAR INTERACTIONS ACROSS TISSUES

By

Fahim Mohammad
B. Tech. (CSE), AMU Aligarh, India, 2001
M. Tech. (IT), IPU Delhi, India, 2007

A Dissertation Approved on

June 18, 2012

by the following Dissertation Committee

Dr. Eric C. Rouchka, Dissertation Director

Dr. Adel S. Elmaghraby

Dr. Ibrahim Imam

Dr. Ming Ouyang

Dr. Jeffrey C. Petruska

DEDICATION

This dissertation is dedicated to my parents

who have given me unconditional love,

selfless support

and

invaluable educational opportunities.

ACKNOWLEDGEMENTS

When I started writing this chapter, I felt that I suddenly became more expressive in comparison to writing the entire thesis. I admit that this dissertation would not have been possible without the constant support and encouragement from a number of individuals. It is to them that I owe my heartfelt gratitude.

First and foremost, I offer my sincerest gratitude to my advisor, Dr. Eric C. Rouchka, for his valuable guidance and constant support throughout my Ph.D. His knowledge and ability to explain things simply and clearly made bioinformatics interesting for me. His perpetual energy and research acumen have motivated me to work diligently. I deeply appreciate his effort and patience to read my thesis over and over again.

His emphasis on overall development of his students is commendable. He always encouraged me to attend conferences and workshops and to participate in poster presentations, research talks and research competitions. He funded my travel to Taiwan for a conference and Los Angeles for a workshop. He also provided me with a stimulating research environment in the form of weekly lab meetings, journal club meetings, bioinformatics retreats, summits and interdisciplinary collaborations. He never pushed me into anything and I must say that I enjoyed the freedom to deal with research problems my own way. At the same time, this freedom also gave me a sense of responsibility to finish my work honestly.

The infrastructure and facilities that he provided for my research was unparalleled. He always provided me a machine (in fact he gave me two high–end machines) with the best configuration available in the market. I came to know him personally when I was accompanying him for a conference in Taiwan. I can confidently say that he is not only a good supervisor but also a very good human being. For me, it was a wonderful experience working with him and honestly, I enjoyed everyday of my Ph.D.. I simply couldn't have wished for a better supervisor.

I am truly indebted to Dr. Jeffrey C. Petruska for allowing me to carryout biological experiments in his lab. He was generous enough to include me in his research team. This thesis work uses his data as a test case for designing a generic approach for solving associated biological problems. I am sure this thesis would have not been possible without his help.

Working as a part of a large systems biology team, I also enjoyed the company of Dr. Robert M. Flight and Dr. Ben Harrison. I would like to acknowledge their support, guidance and helpful discussions throughout this work. Robert's experience, insight and suggestion during my work helped me refine it. His editing suggestions and precise sense of language contributed to the final copy of this dissertation. Ben was extremely helpful in explaining biological problems, laboratory procedures and data. I enjoyed working with Ben, performing PCR experiments in Dr. Petruska's lab.

Dr. Elmaghraby, Dr. Imam and Dr. Ouyang deserve special thanks for being part of my dissertation committee. In particular, I would like to thank Dr. Elmaghraby, Chairman of the Department of Computer Engineering and Computer Science, for his constant support and encouragement.

I am also grateful to all the faculty members of the Department of Computer Engineering and Computer Science for their support especially Dr. Desoky, Dr. Kantardzic and Dr. Nasraoui. Dr. Kantardzic is the one who taught me data mining and I admit that I learned a lot from this class. I also worked with him briefly which broadened my research experience.

Collective and individual acknowledgments are also owed to my lab members, Dazhuo Li, Ernur Saka and Abdallah Eteleeb for helping me in different capacities. For almost three years, I enjoyed working in the lab as a close–knit family. We use to have discussions about research, travel, country, culture and food.

It was almost impossible to come to Louisville to pursue my Ph.D. without the financial support from the Department of Computer Engineering and Computer Science in the form of the Grosscurth fellowship. I am indebted to the fellowship committee for their decision in my favor. This fellowship lasted two years (2008 – 2010). I am also thankful to Dr. Rouchka, Dr. Nigel Cooper and KBRIN

(Kentucky Biomedical Infrastructure Network) for funding my Ph. D. for almost two years (2010 – till date).

I wish to thank my colleagues and friends back in India for their continuous support, specially Prof. M. N. Doja, Prof. Tanvir Ahmad and Sohrab. Prof. M. N. Doja is a father–like figure and all time mentor for me. When I had a choice of selecting between Indian Institute of Technology (IIT), Kanpur and University of Louisville for pursuing Ph. D., he was the one who suggested me to come to Louisville. Now, I feel that coming here was my right decision. Prof. Tanvir Ahmad is one of my best friends and big brother, who is always ready to help. He is like panacea for me. Sohrab is my school time friend and has always been incredibly supportive. I treat these wonderful persons as rare gifts from God.

My appreciation also goes to all my brothers, sisters, family members and friends their unequivocal support throughout.

Words are not enough to express my appreciation for my better half, my wife, Saba, for supporting and encouraging me to complete this degree. I would like to thank her for understanding, consideration and unconditional love. I am also thankful to my children Ammar and Abaan, whose smiles and innocence always invigorate me and bring happiness in my life.

My final acknowledgment goes to my parents who bore me, loved me, educated me and then freed me to realize my dreams. I appreciate their unconditional love for me, remembering me in their prayers, supporting me to obtain the best education and dealing with all of my absence from many family occasions with a smile. To them I dedicate this dissertation.

— Fahim Mohammad

ABSTRACT

A SYSTEMS–BASED APPROACH FOR DETECTING MOLECULAR INTERACTIONS ACROSS TISSUES

Fahim Mohammad

June 18, 2012

Current high–throughput gene expression experiments have a straightforward design of examining the gene expression of one group or condition relative to that of another. The data is typically analyzed as if they represent strictly intracellular events, and often treats genes as coming from a homogeneous population. Although intracellular events are crucial to nearly all biological processes, cell–cell interactions are often just as important, especially when gene expression data is generated from heterogeneous cell populations, such as from whole tissues. Cell–cell molecular interactions are generally lost in the available analytical procedures and as a result, are not examined experimentally, at least not accurately or with efficiency. Most importantly, this imposes major limitations when studying gene expression changes in multiple samples that interact with one another.

In order to addresses the limitations of current techniques, we have developed a novel systems–based approach that expands the traditional analysis of gene expression in two stages. This includes a novel sequence–based meta–analytic tool, *AbsIDconvert*, that allows for conversion of annotated features using an interval tree for storing and querying absolute genomic coordinates for comparison of multi–scale macro–molecule identifiers across platforms and/or organisms.

In addition, a systems–based heuristic algorithm is developed to find intercellular interactions between two sets of genes, potentially from different tissues by utilizing location information of each gene along with the information available in the secondary databases in the form of interactions, pathways and signaling.

AbsIDconvert is shown to provide a high accuracy in identifier conversion as compared to other available methodologies (typically at an average rate of 84%) while maintaining a higher efficiency ($O(n * \log(n))$. Our intercellular interaction approach and underlying visualization shows promise in allowing researchers to uncover novel signaling pathways in an intercellular fashion that to this point has not been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Current trends in high–throughput gene expression analysis

Typical high–throughput experiments have a straightforward design of examining the gene expression of one group/condition relative to that of other group such as control vs treated or healthy vs diseased. Advancement and sophistication in the primary and secondary analytical tools have made these analysis reliable, accurate and rapid and have enabled the user to extract greater meaning from large datasets in the form of statistically over–represented signaling pathways, genes following similar expression patterns, and clustering and classification of samples and/or genes. However, analytical tools currently employed generally examine the data as if they represent strongly intra–cellular events, and often treat them as coming from a homogeneous cell population.

1.2 Motivation

Although intracellular events are crucial to nearly all biological processes, cell–cell interactions are often just as important, especially when gene expression data is generated from a heterogeneous cell population, such as from whole tissue. Cell–cell interactions are generally lost in the available analytical procedures and as a result, are not examined experimentally, at least not accurately or with efficiency. Most importantly, this imposes major limitations when studying gene expression changes in multiple samples that interact with one another. Examples of such interactions include migratory processes (e.g., immune cell transvascular migration, nervous system development, and cancer metastasis), binding processes (e.g., oocyte implantation and leukocyte tethering and rolling) induction processes (e.g., stem cell generation and floor–plate or roof–plate modulation of neuronal

fate) and plasticity processes (e.g., neovascularization, axonal regeneration or sprouting, and sequestration of cancerous or infected cells.)

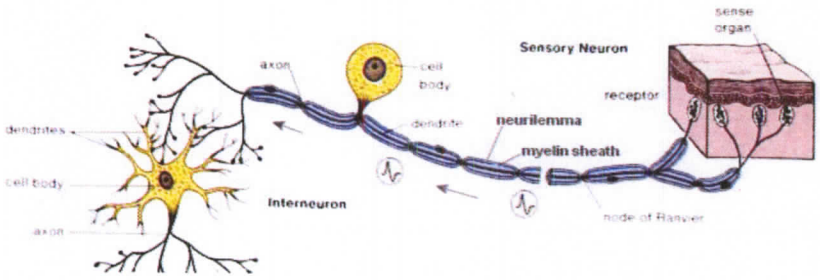

Figure 1.1: Information transfer along sensory neuron [1]

Neurons interact with and are influenced by multiple tissues. For example, in the peripheral nervous system, sensory neurons innervate the spinal cord and skin, while the cell body is located within the ganglion. Electrical and chemical (anterograde/retrograde) signaling modulates the molecular biology of the three distinct biological compartments (spinal cord/ganglion/skin) (Fig. 1.1). Technical limitations dictate that each tissue must be profiled independently, yet these multiple profiles must be considered together in order to address the systems-level aspects of the overall process. Current bioinformatics tools are not equipped to consider profiles from multiple samples (here, tissues) that interact within one system.

1.3 Specific aims

To address the above limitations, a systems–based approach is developed and implemented to analyze high–throughput gene expression data. This approach is heuristic and utilizes data available in the publicly available databases. There are two major components of this approach:

1. *AbsIDconvert* is a meta–analytic tool for converting a set of gene identifiers, genomic sequences or intervals into target identifiers. This sequence–based conversion is accomplished by: 1. mapping sequences of all available genomic identifiers onto their genome and finding their mapping locations, 2. storing these locations into an interval tree, and 3. querying these tree to find target identifiers.

2. A heuristic algorithm that generates interaction paths between two sets of genes. Considering these two sets of genes possibly from two tissues as seeds and using protein interaction information available in publicly available databases, this algorithm finds all interacting genes between two tissues. While finding interactions, it also uses location information of involved genes and removes any gene that is irrelevant in order to keep the final set of interactions minimal.

Development of this approach is driven by a specific problem in neurobiology, namely identification of the genes regulating the neuronal plasticity process of axonal collateral sprouting (where existing intact axons extend new branches and functional connections). It is known that the process of collateral sprouting (CS) involves significant interaction between the neuron undergoing plasticity and the target tissue which is generally other neurons or a peripheral tissue (i.e. involves intercellular, inter-tissue interactions). Therefore gene expression datasets are generated by *Dr. Jeff Petruska's Lab* for *sensory neurons undergoing CS*, *their peripheral target tissue (skin)*, and *their nervous system target tissue (spinal cord)*. The genetic control of this process is coordinated across multiple interacting tissues. Current molecular informatics tools fall far short of allowing an efficient analysis of this interplay and uncovering the many signaling aspects that control this process. Although the tool is developed in the framework of the model of CS, the analysis is readily applicable to any experimental design in which a separation can be achieved for two or more interacting tissues, cell populations or potentially host–pathogen relationships.

The overview of chapters 2 through 6 follows:

Chapter 2 briefly explains the relevant basic molecular biology. It also explains the Central Dogma of Molecular Biology and the process of transcription and translation that are necessary for every living organism. Recent advances in molecular biology techniques have resulted in a number of high–throughput sequencing methods including next–generation sequencing and microarrays. These sequencing tools are capable of sequencing the complete genome of living organisms. This thesis work uses data generated from these technologies which are briefly explained. Genomes and sequencing approaches are discussed in sections 2.2 and 2.3. Genome assembly, alignments, annotations and

annotation databases are covered in separate sections. Microarrays are briefly discussed in section 2.7.

Chapter 3 explains the algorithms used in the construction of AbsIDconvert. The 2012 database issue of Nucleic Acid Research reports a total of 1380 databases covering various areas of molecular biology [2]. Most of these databases are independent of each other and annotate genetic entities differently. This large collection of heterogeneous datasets results in issues with the storage and comparison of annotations across entities. This chapter focuses on this concern and proposes an interval–tree as an efficient means for the storage, search and maintenance of genetic entities. Section 3.1 introduces to the problem and describes how these annotations can be represented as intervals. Section 3.2 discusses intervals and the criteria for detecting overlapping intervals in a system with multiple intervals. Section 3.3 describes interval–trees, a data structure to store annotations (as intervals) and perform associated operations. Section 3.4 describes the design steps in finding overlapping annotations. It also briefly describes two alignment algorithms, which will be used later in our approach to map sequences onto a reference genome. Results are shown in section 3.5. Finally, section 3.6 concludes this chapter.

Chapter 4 describes *AbsIDconvert*, a tool for comparing multi-scale macromolecule identifiers across platforms/organisms/labs to allow for powerful meta–analyses. Meta–annotations are extremely dynamic and change on a daily basis. Rather than relying on different meta–analysis databases, AbsIDconvert is constructed for mapping between various annotation granularities at the locus, transcript, sequence, and probe level. The key to this novel system is to reduce each identifiers to the sequence level which is common between all annotations. For organisms with a reference genome available, each annotation can be aligned to the respective genome and given absolute coordinates. Depending on the alignment positions on the genome, interval information for each identifier is found and maintained in an interval tree. These interval trees can then be queried to find all overlapping identifiers for a particular identifier. AbsIDconvert has many potential uses, including gene identifier conversion and cross-species comparisons. AbsIDconvert provides an efficient, accurate and reliable mechanism for conversion between two identifier domains of interest.

The flexibility of AbsIDconvert will allow for these identifier domains to be custom defined as long as a genomic mapping interval can be determined.

This chapter discusses all aspects of AbsIDconvert. Section 4.1 introduces the problem of identifier (ID) conversion, associated problems and challenges. Section 4.2 describes the available methods and tools to perform ID conversion. The next section 4.3 describes the drawbacks associated with the available tools. Section 4.4 describes the AbsIDconvert approach for performing ID conversion. System design and implementation is discussed in section 4.5. Section 4.6 reports the results and includes run time and output comparison of AbsIDconvert with a number of available tools. Section 4.7 details three case studies to show the applicability of AbsIDconvert which is otherwise difficult. Section 4.8 concludes this chapter.

As mentioned previously, cell–cell interactions are as important as intracellular interactions when gene expression data is generated from a heterogeneous cell population, such as from whole tissue. Chapter 5 discusses a heuristic algorithm for detecting intercellular interactions from two sets of genes. The heterogeneous gene sets can be preprocessed using AbsIDconvert to make the data compatible for comparison. Section 5.1 introduces the actual problem of detecting intercellular interactions. Section 5.2 lists and explains some of the available protein interaction databases. Section 5.3 briefly introduces some of the available algorithms to find intercellular interactions. Section 5.4 discuss the methodology and design. Section 5.5 analyses the fitness and completeness of the algorithm. Section 5.6 reports the results. Section 5.7 concludes this chapter.

Chapter 6 is dedicated to summary and future work.

CHAPTER 2

BACKGROUND

2.1 Basic molecular biology

2.1.1 Organism and cells

All living organisms are composed of small cells, often too small to be seen by a naked eye. These cells are the basic structural and functional unit of all known organisms and often termed as the building block of life. A typical cell size ranges from 1 μm in bacteria to 100 μm in plant. The estimated number of cells in the human body is more than 60 trillion and there are roughly 320 different cell types [3]. Organisms may be categorized as unicellular or multicellular based on whether they are composed of a single or multiple cells. Another categorization may be based on the presence or absence of a nucleus in their cells. *Prokaryotes* lack a nucleus and their DNA (explained later) floats loosely in the liquid center of the cell (Fig. 2.1). Prokaryotic organisms were the only form of life millions of years ago, and they gradually evolved into complex organisms. *Prokaryotes* are unicellular organisms while *eukaryotes* are composed of both unicellular and multicellular organisms with a well-defined nucleus to house their DNA.

2.1.2 Chromosomes

A chromosome is a thread like structure with a single piece of coiled DNA. It may contain proteins, which serve to package the DNA and control their functions. Prokaryotes have a single circular hoop–shaped DNA whereas eukaryotes have one or more chromosomes housed in the nucleus. Eukaryotic chromosomes are long strands of DNA tightly wound around proteins into a condensed structure called *chromatin*. In humans, there are 22 pairs of autosomal chromosomes and a pair of sex

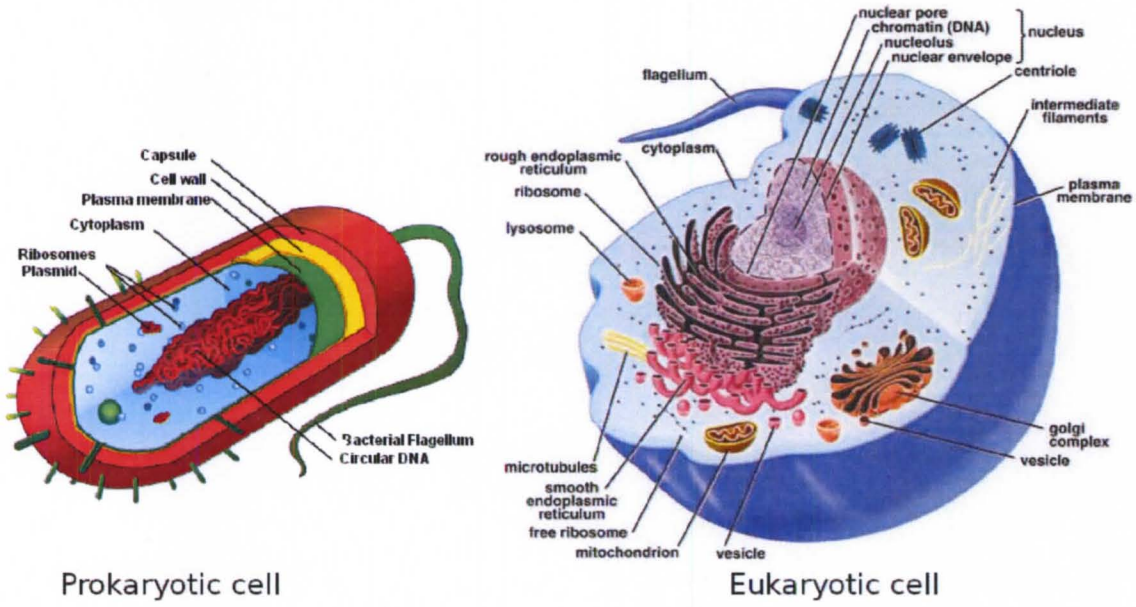

Figure 2.1: Prokaryotic and eukaryotic cell [4]. Used with permission.

chromosomes. In each pair, one chromosome is inherited from father and the another from mother. The sex chromosomes are X and Y determine the sex of a human being. Females have two X chromosomes whereas males have an X and a Y chromosome [5] [6]. For organisms to grow, reproduce and pass genetic information, these chromosomes must be copied and divided in a regulated manner.

2.1.3 Molecules of life

There are four categories of molecules important for a life: small molecules, nucleic acids (DNA and RNA) and proteins. DNA, RNA and proteins are collectively termed as biological macromolecules. Small molecules are the building blocks for macromolecules and may be involved in functions such as signal transmission, biochemical reactions and cellular processes. Examples include water, amino acids, nucleotides, sugars and some fatty acids [4].

DNA

Every living organism on earth uses *D*eoxyribo*N*ucleic *A*cid (DNA) to store and pass genetic information from one generation to the next. DNA is necessary for the development and functioning of all living organisms. During the 1920s, *P.A. Levene* analyzed the components of the DNA molecule

and concluded that DNA contains four nitrogenous bases: adenine(A), guanine(G), cytosine(C) and thymine(T); deoxyribose sugar; and a phosphate group [4]. These nitrogenous bases can be classified into two types: purines and pyrimidines. Purines have two fused rings with two nitrogen atoms within each ring whereas purines have a single–ring structure with two nitrogen atoms within the ring (Fig. 2.2).

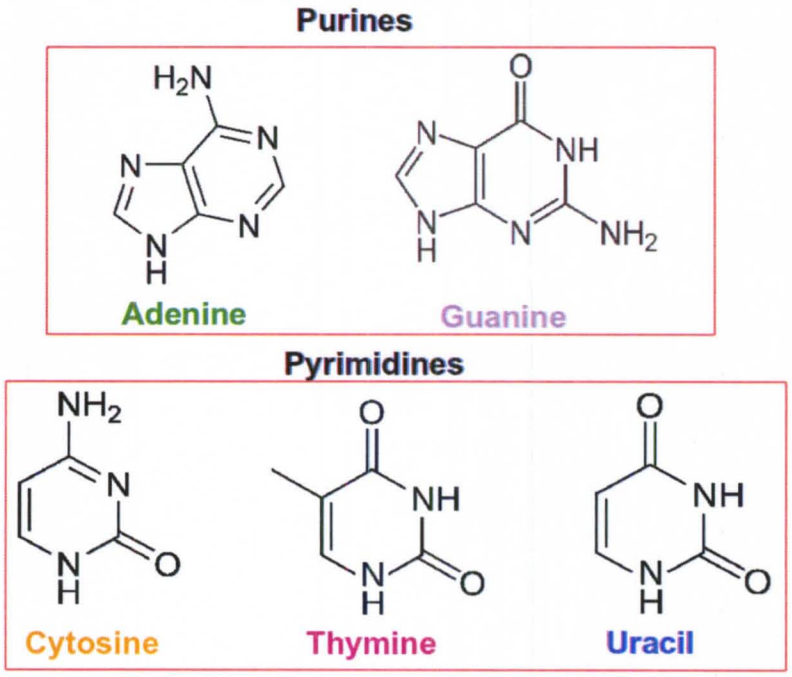

Figure 2.2: Chemical structure of the nucleotides.

In 1953, James D. Watson and Francis H. C. Crick at Cavendish Lab in Cambridge solved the mystery of the structure of DNA by proposing a simple double helix model which earned them the Nobel Prize in 1962 [8] [9]. DNA consists of two long polymers of nucleotides (polynucleotides) with backbones made of sugars and phosphate groups joined by ester bonds (Fig: 2.3). These two polymers which may be of any length and contain any sequence, run in opposite directions of each other and are therefore anti-parallel. The opposite strands stick together via two hydrogen bonds between A and T, and three hydrogen bonds between C and G, forming a ladder–like structure [9]. These hydrogen bonds are individually weak but collectively quite strong that makes double helix DNA stable [5]. The two ends of the strands are chemically different and thus, a 5' or 3' directionality can be assigned to each polynucleotide based on the carbon atoms of the sugar molecule. The

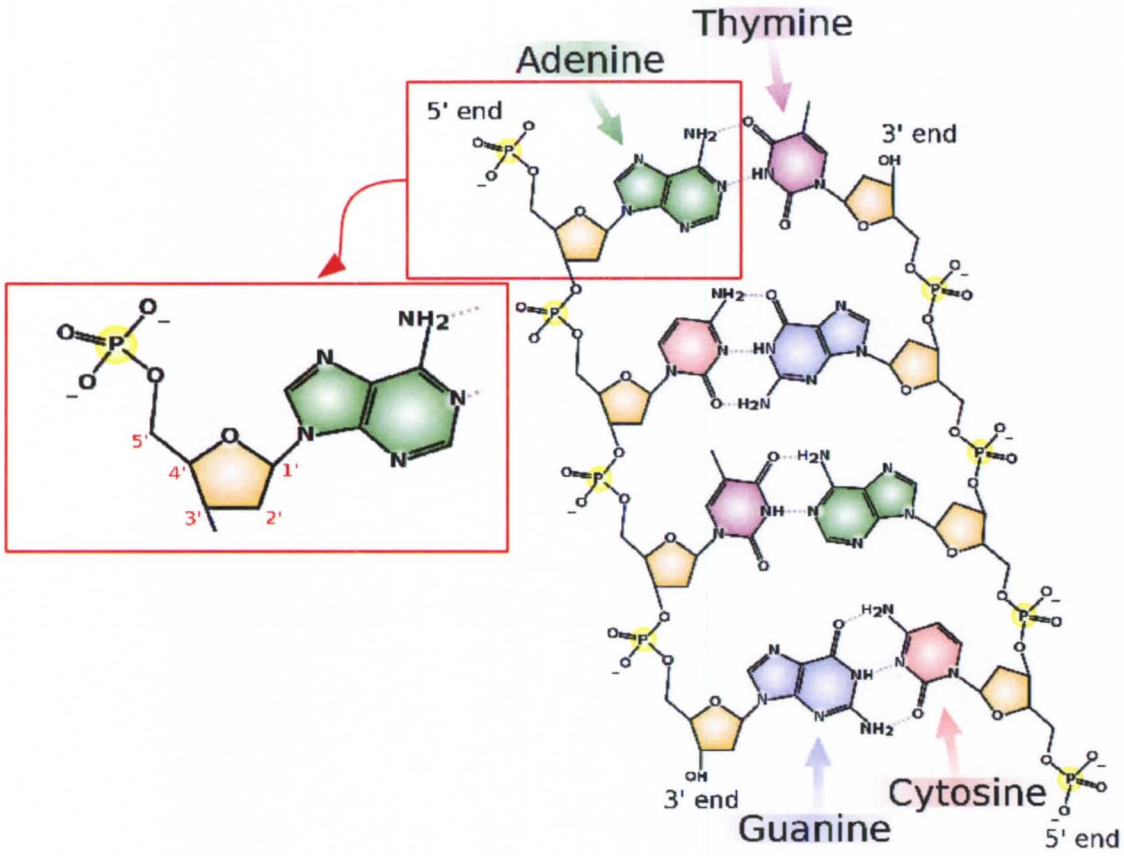

Figure 2.3: Chemical structure of a DNA molecule [7]. Numbers in inset shows how the carbon atoms are numbered in a sugar molecule. Used with permission.

polynumcleotide sequence in Fig. 2.3 is ACTG. The length of a DNA molecule is usually measured in base-pairs (bp) or nucleotides (nt).

DNA replication is the basis for biological inheritance and is a mechanism in which one double–stranded DNA is replicated into two identical ones. The DNA double helix unwinds and forks during this process, and a new complimentary strand is synthesized by specific molecular machinery on each branch of the fork (Fig. 2.4). This happens during cell division and a copy of the original goes to the newly formed daughter cells [4] [11].

RNA

*Ribo*N*ucleic* A*cid* (RNA) is similar to DNA except that *Thymine (T)* is replaced by *Uracil (U)*. In addition, RNA nucleotides have the sugar ribose incorporated whereas DNA nucleotides use

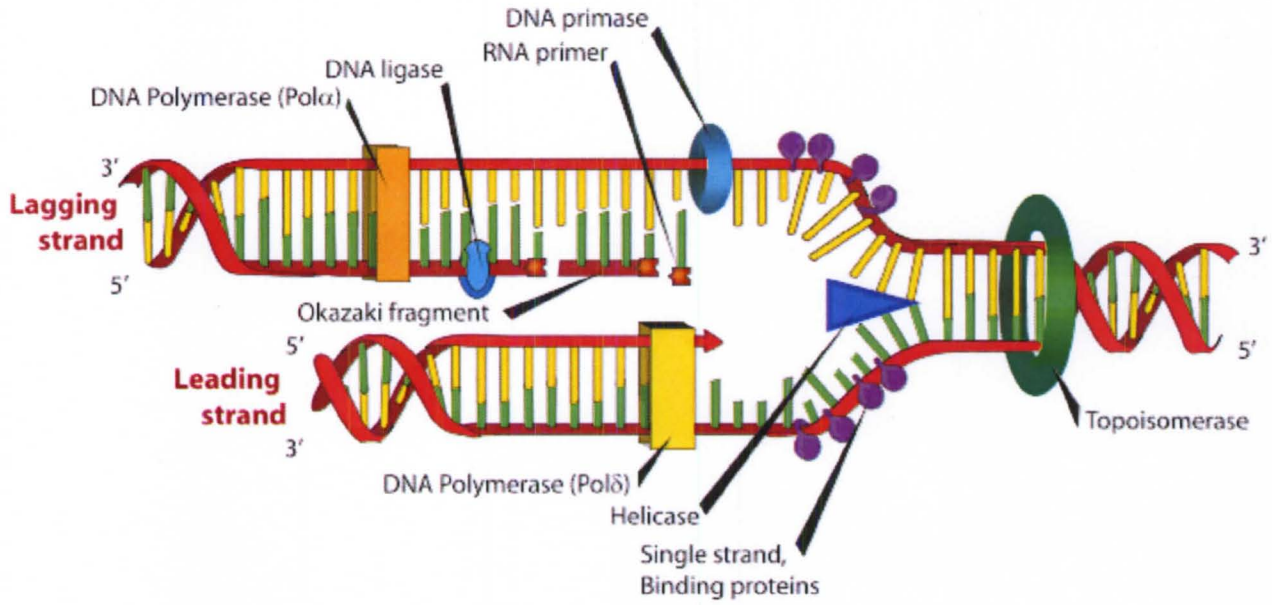

Figure 2.4: DNA replication [10]

deoxyribose. RNA molecules typically consist of a much shorter chain of nucleotides and are less stable than DNA. RNA can be single stranded or double stranded, but is generally found in single–strand form. Most biologically active RNAs, including mRNA, tRNA, rRNA, snRNAs and other non-coding RNAs, contain self-complementary sequences that allow parts of the RNA to fold and pair with itself to form double helices.

mRNA

Messenger RNA (mRNA) encodes genetic information transcribed from a DNA template into a series of three–base codons, each of which specifies a particular amino acid with the exception of stop codons, which terminate protein synthesis. The mRNA carries this genetic information into the cytoplasm where protein synthesis occur.

miRNA

MicroRNA (miRNA) are naturally occurring small (22 nt) non–coding RNA usually found in eukaryotic cells. MiRNA are post–transcriptional regulators and may bind to mRNA molecules resulting

in downregulation of gene expression through translational repression, mRNA cleavage and deadenylation.

Proteins

Proteins, dubbed as *workers in the cellular factory*, are responsible for carrying out many functions of the cell, including metabolism, transport, communication, structure and division. Proteins are sometimes also touted as the "movers and shakers" of the cell– whatever is the job, they get it done. They interact with other molecules to carry out their functions. Proteins begin as polymers of amino acids, called *polypeptides*. A protein becomes functional when it is folded. The size of the protein molecule can vary from a few to thousands of amino acids in length. For example, *insulin* is a small protein with 51 amino acids whereas *titin* has $\approx 28,000$ amino acids [4]. The shapes of the proteins are complex and essential for function and may vary from primary structure to quaternary structure such as *hemoglobin proteins* (Fig. 2.5) [11].

2.1.4 Central Dogma of Molecular Biology

> *"I just didn't know what* **dogma** *meant. And I could just as well have called it the 'Central Hypothesis', or – you know. Which is what I meant to say. Dogma was just a catch phrase. [8]"*
>
> —Francis Crick

The Central Dogma of Molecular Biology explains information transfer from genotype to phenotype and states that, once an information (sequences) get into protein, it cannot get out again [13]. It classifies a total of nine possible information transfers into three groups each containing three types of transfers. *General transfers* are believed to take place in most cells and include DNA $\rightarrow$ DNA (replication), DNA $\rightarrow$ RNA (transcription) and RNA $\rightarrow$ protein (translation) transfers (Fig. 2.6). DNA replication is a biological process in which a DNA molecule is copied. DNA transcription involves the transcribing of the genetic information from DNA to mRNA. In translation, the mRNA, produced during transcription, is decoded by the ribosome to produce a specific amino acid chain, that will later fold into an active protein. *Special transfers* are the cases which are known to occur

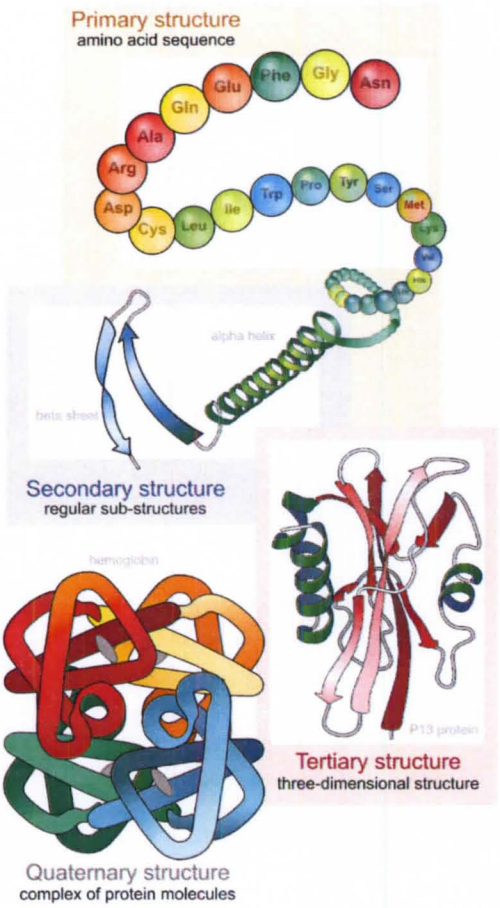

Figure 2.5: Levels of protein structure [12]. Used with permission.

only under specific conditions. Examples include RNA → RNA (RNA replication), RNA → DNA (Reverse transcription) and DNA → protein. Reverse transcription is the information transfer from RNA to DNA and are known to occur in the case of retroviruses such as HIV. Direct translation from DNA to protein has been demonstrated in laboratory setup (in vitro). The last group is *unknown transfers*, which are not known to occur, includes protein → protein, protein → DNA and protein → RNA [14] transfers.

2.1.5 Gene

A gene is a fragment of genomic DNA that can be transcribed into an mRNA sequence that is subsequently translated into a protein. It is a molecular unit of heredity of all living organisms and holds information to build and maintain an organism's cells and pass genetic traits to the next

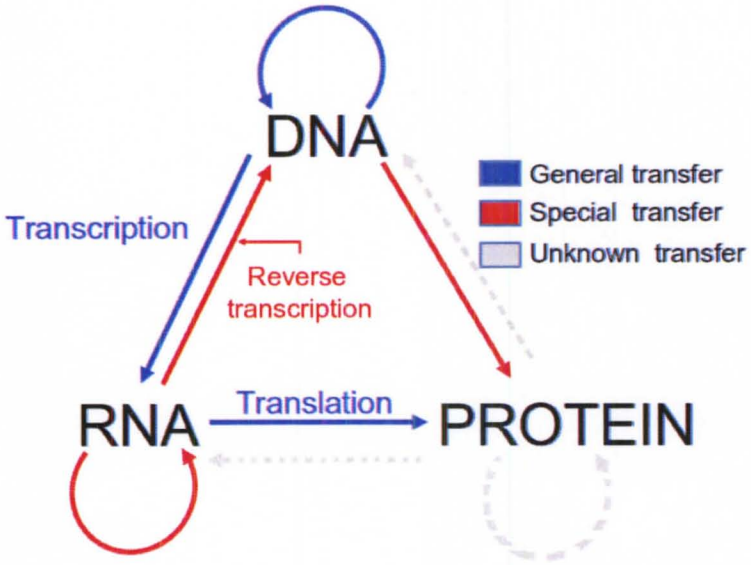

Figure 2.6: Central Dogma of Molecular Biology

generation. The total number of human genes was initially estimated to be around 100,000. The draft genome sequence paper [15] published in Feb, 2001 estimated only about 30,000 to 40,000. Although the exact number of human genes is still unknown, researchers estimate it to be fewer than 30,000. In eukaryotic genomes, the coding portion of a gene, called exons, are interrupted by intervening sequences, called introns. Both exons and introns are transcribed into pre–mRNA. Promoters and enhancers determine what portions of the DNA will be transcribed into the precursor mRNA (pre–mRNA). The exons in the pre–mRNA are spliced together to form a mature mRNA, which is later translated into protein (Fig. 2.7).

2.1.6 Transcription

Transcription is the process of creating a complementary RNA copy of a sequence of DNA. This process is accomplished in three steps: initiation, elongation and termination. During *initiation*, RNA polymerase binds at a sequence called a *promoter*. A typical promoter sequence in many eukaryotes is TATA box as its sequence consists of TATAAA (Fig. 2.8). A promoter tells the RNA polymerase that the gene to transcribe is about 30 base pairs downstream. Transcription is performed on the template strand and the resultant RNA is the transcript of the nontemplate

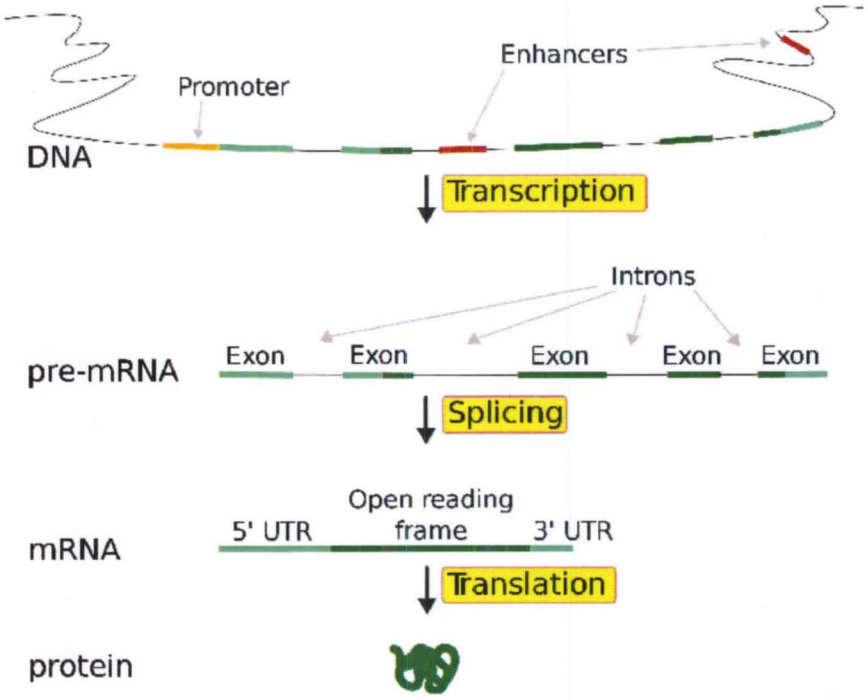

Figure 2.7: Eukaryotic protein–coding gene. Used with permission.

strand. RNA polymerase along with other proteins, called transcription factors, opens up the DNA double helix and start reading the template strand in a 3' to 5' direction. In *elongation*, the RNA polymerase traverses the template strand and produces an RNA copy from 5' to 3' direction. This RNA molecule is an exact copy of the nontemplate strand except that *thymines* replace by *uracils*. In *termination*, the RNA polymerase encounters the terminator sequence and transcription stops at this place. At this time, the mRNA gets detached from the template and the double–stranded DNA molecule snaps back into its natural helical shape.

2.1.7 Post–transcription process

After being produced, the transcribed RNA (precursor mRNA or pre–mRNA) goes through some additional modification in eukaryotes including capping, polyadenylation and splicing. During *capping*, a 5' cap is added to the mRNA that helps in ribosomal binding during translation. In *polyadenylation*, a long string of adenines are added to the 3' end of the pre–mRNA. This string is also sometimes referred as *poly–A tail*. A poly–A tail increases the half-life of mRNA and also helps in increased

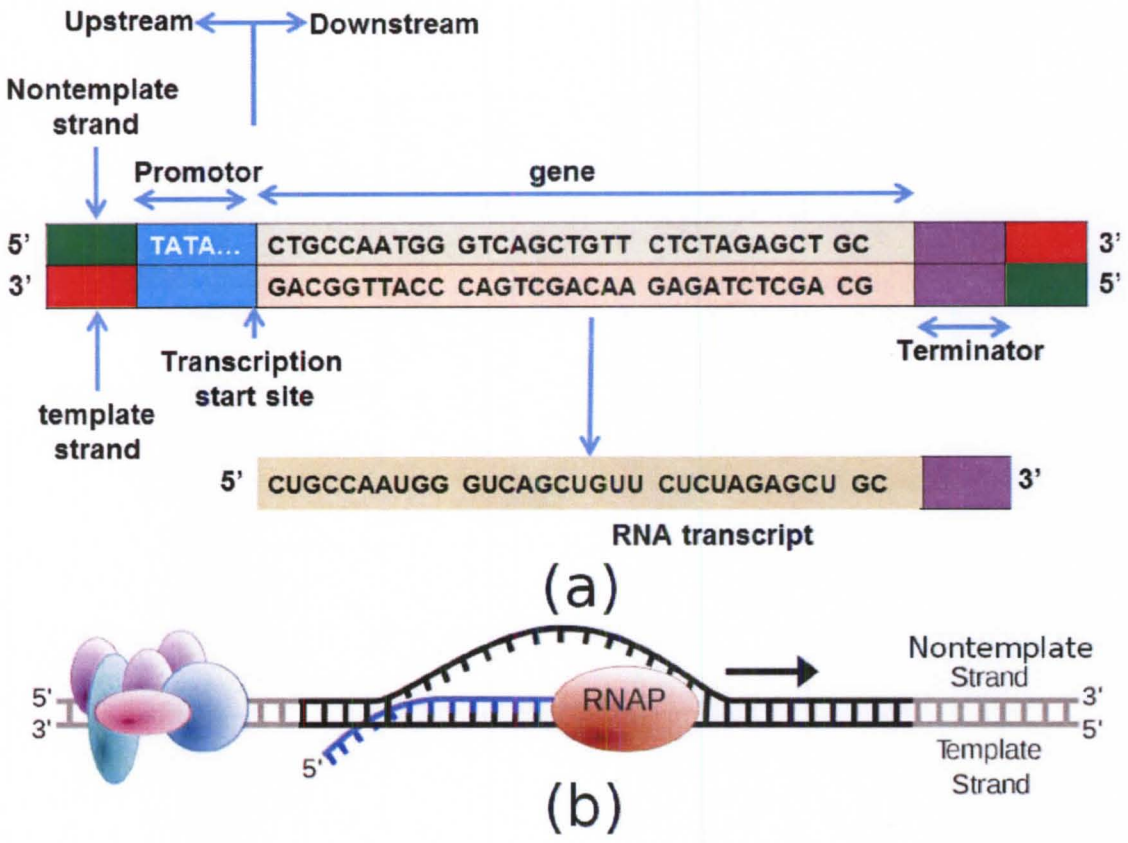

Figure 2.8: (a) The transcription unit. (b) Elongation process during transcription.

translation. *Splicing* removes introns, the noncoding region, from the pre–mRNA and stitches the exons together without interruption. Once post–transcriptional processing is complete, the mRNA migrates out of the cell nucleus, into the cytoplasm where it is translated into a protein.

Alternative splicing is a process through which exons in the pre–mRNA are spliced together in multiple ways to form a mature mRNA. The resulting mRNA may be translated into different protein isoforms; thus, a single gene may code for multiple proteins. Fig. 2.9 represents four common types of alternative splicing. In type (a), different promoters may be used for different splice variants which result into mRNA transcripts having different start sites. Type (b) represents selection of different poly–A sites that result in different 3' ends. An entire exon may be skipped in this process. In the third type (c), introns may be retained in the final transcript. In type (d), an entire exon or a combination of exons may be skipped to form different transcripts.

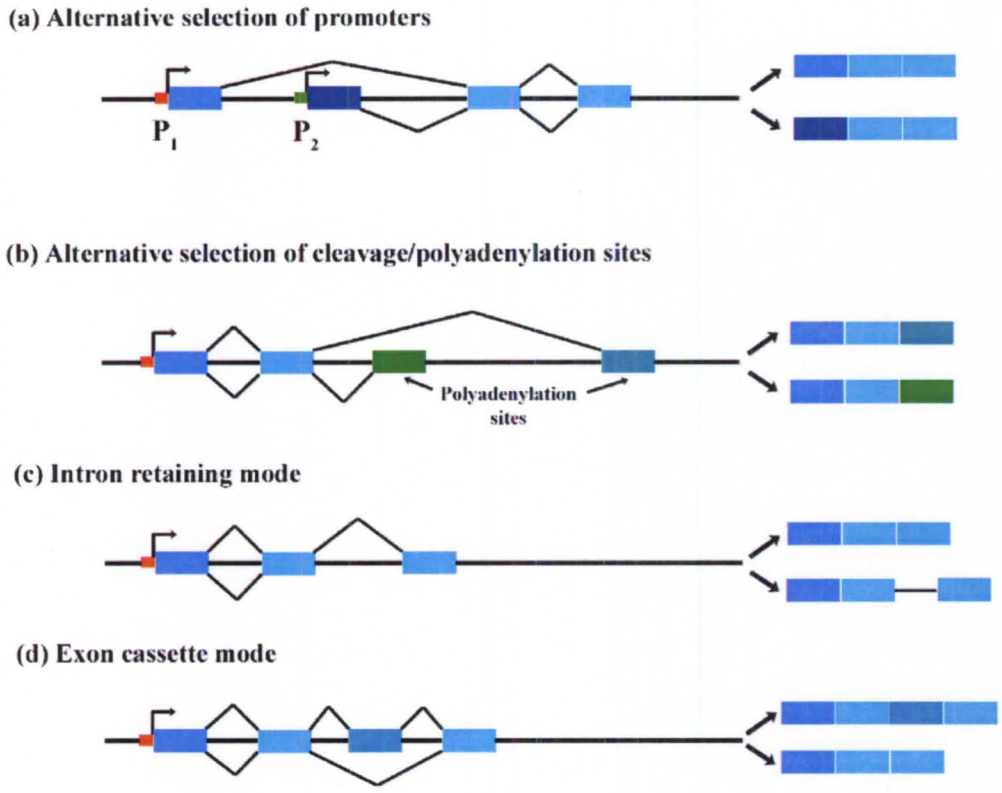

Figure 2.9: Common types of alternative splicing [16]

2.1.8 Genetic code

The genetic code is the set of rules by which information encoded in genetic material is translated into amino acid sequences. The code defines a mapping between tri-nucleotide sequences, called codons, and amino acids. There are 64 different codons that result in 20 amino acids, thus resulting in degeneracy, with more than one triplet coding an amino acid. In most cases, the first and second base of the triplets coding for a particular amino acid remain same with the difference in the third or wobble base. The start codon called *methionine* is coded by AUG. The stop codons are UAA, UAG, and UGA and do not encode any amino acid. The stretch of codons between AUG and a stop codon is called an *open reading frame (ORF)* [17](Fig: 2.10).

Given an mRNA sequence, translation to the corresponding amino acid may start either at first, second or third base of an oligonucleotide. Considering a double–stranded DNA sequence, there are

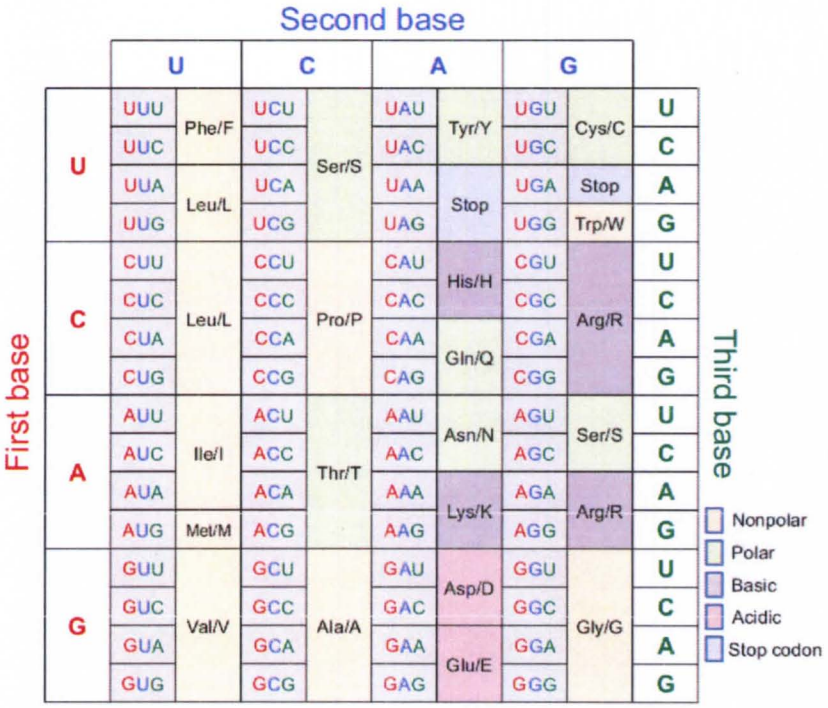

Figure 2.10: Codons to amino acid conversion

three translation start sites possible on each strand. Each of these frames may produce a completely different amino acid sequence. An example of this conversion on one strand is shown in Fig. 2.11.

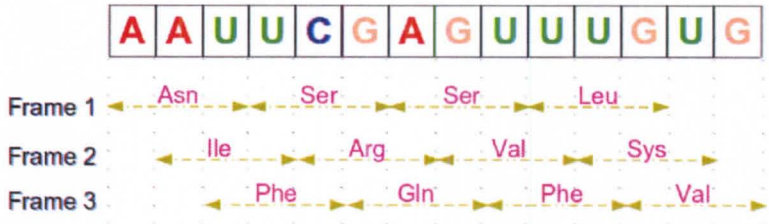

Figure 2.11: Translation of a single stranded mRNA into an amino acid sequence.

2.1.9 Translation

Once transported into the cytoplasm, an mRNA can be translated into a polypeptide using the genetic code with the help of ribosomes, tRNA, rRNA and other components.

- *Transfer RNA* (tRNA) is an adapter molecule composed of RNA, typically 73 to 93 nucleotides in length. These are produced by transcription but are never translated. It has a unique

three dimensional structure and carries an amino acid molecule on one end and a three–letter anticodon on the other end. Its job is to bring the amino acid molecule to the ribosome and help in translation.

- In the cytoplasm, *ribosomal RNA* (rRNA) combines with proteins to form ribosomes, which act as a site of protein synthesis.

- A ribosome is a large complex molecule which is responsible for catalyzing the formation of proteins. Ribosomes are found in all living organisms. They are made up of two subunits: large and small, which have their own rRNA, and are capable of constructing any sort of protein. A ribosome has three binding sites; 1) *A* site, where tRNA inserts its anticodon arm to match with codon of the mRNA molecule, 2) *P* site, where amino acids are attached using peptide bonds and 3) *E* site, where tRNAs are released from the ribosome after their amino acids become part of the growing polypeptide chain.

Translation proceeds in three steps: initiation, elongation and termination.

- The *initiation* starts with binding the small subunit of a ribosome to the 5' end of an mRNA. This subunit proceeds downstream until it encounters the start codon where it is joined by the large subunit. A tRNA with an anticodone sequence identical to the complementary mRNA codon binds at the P site of the ribosome [18].

- During *elongation*, the ribosome calls for the tRNA carrying the amino acid specified by the codon residing in the A-site. An appropriate tRNA is able to base pair with the next codon on the mRNA. The preceding amino acid bonds with the incoming amino acid via a peptide bond. Once the bonding is complete, the ribosome shifts to the next codon on mRNA (this shifting is called *translocation*). The initiator tRNA then moves to E site and is later released. This process is repeated until all the codons in the mRNA has been read by tRNA [19].

- Once the ribosome reach a stop codon (UAA, UAG and UGA), no more amino acids can be added. In place of tRNAs, another protein called release factors, bind to the ribosome.

This binding initiates the cleavage of the polypeptide chain and release of the subunits of the ribosome.

The polypeptides are then folded into one or more specific spatial conformations, driven by a number of non-covalent interactions which then carry out a variety of biological functions.

2.1.10 Untranslated regions (UTR)

During translation, the regions those are not translated include the cap, the 5' UTR, 3' UTR and poly–A tail (Fig. 2.12). Five–prime (5') UTRs may contain regulatory elements that can positively control gene expression. In prokaryotes, the 5' UTR usually contains a ribosome binding site (RBS), also known as the Shine Dalgarno sequence (AGGAGGU). The median length of 5' UTRs is approximately 150 nt but may be as long as several thousand bases [18].

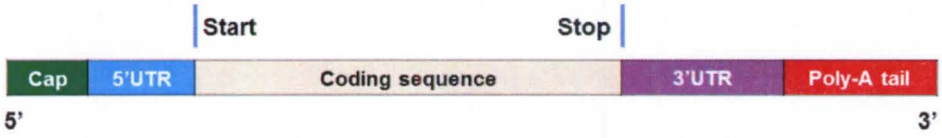

Figure 2.12: Mature mRNA structure including the UTR regions.

The three prime (3') UTR is found on the 3' side of mRNA and after the stop codon. Several regulatory sequences may be found on 3' UTR responsible for affecting the stability of proteins and their cellular localization including miRNAs binding sites, cytoplasmic polyadenylation elements and zipcode binding domains [20].

2.2 Genomes

A genome is the total amount of genetic information contained in the chromosomes of an organism and is encoded either in the form of DNA or, for many viruses, as RNA. The genome size for organisms vary: ranging from a few kilobases (viruses) to tens of gigabases (human and fish). A genome includes both the coding as well as the non-coding sequences of the DNA/RNA. Out of 3.2 billion DNA base pairs in the human genome, only about 1.5% code for proteins while the rest

consist of non–coding genes, regulatory sequences, UTRs, introns, repetitive elements, and intergenic regions (`http://www.ebi.ac.uk/2can/disease/genes12.html`).

2.3 Genome sequencing

Genome sequencing is the process of determining the sequence and order of DNA nucleotides in a genome. Almost any biological sample, including saliva, hair, bone marrow, seeds and leaves, can provide the genetic material necessary for sequencing. Genome sequencing can be used as a valuable source for finding genes and proteins, their locations, functions, regulations, chromosomal structures and evolution.

Genome sequencing approaches

Current genome sequencing is not capable of sequencing a complete genome as a single molecule. An alternative method is to fragment a genome into small pieces and then use a sequencing method to find the actual genomic sequence for individual pieces and finally combine these sequences to get the whole genome. In a *clone–based* sequencing approach, a genome is broken into relatively large chunks, called clones, about 150,000 base pairs (bp) long. Several copies of a clone are then selected and fragmented into smaller random pieces ($\approx$ 500 bp) using chemical shearing or sonication [21] which are sequenced individually. Each of these fragment sequences are then assembled based on sequencing overlaps to reconstruct the sequence of the whole clone. The *whole-genome shotgun* approach involves fragmenting the whole genome, sequencing the fragments, and reassembling them into the full genome sequence. This approach is much faster but complicates the assembly process. While the clone–based method produces a much more accurate and complete genome, shotgun sequencing is more prevalent due to the greatly reduced cost and the presence of reference genomes that can greatly facilitate the assembly. Both approaches have been used in whole genome sequencing. The human genome was sequenced using a combination of these two approaches [22].

First–generation sequencing

The Maxim–Gilbert (1977) method of DNA sequencing is based on chemical modification of DNA and subsequent cleavage at specific bases [23]. This method requires the radioactive labeling of the 5' end of DNA and purification of the DNA to be sequenced. Although fairly accurate and popular at that time, this method was complex and difficult as it required strand separation before sequencing. Additionally, it was also considered unsafe because of the extensive use of toxic chemicals.

Sanger and Clouson (1975) used a "Plus and Minus" method to sequence ϕX174 bacteriophage, the first genome [24]. However, this method was limited by its inability to sequence a double stranded DNA molecule. Also, this method required both the "plus" and the "minus" strand to determine the actual sequence. Sanger modified this technique in 1977 and introduced "chain terminator sequencing" that is based on the use of dideoxynucleotides triphosphate (ddNTP) in addition to the normal nucleotides (NTPs) [25]. Dideoxynucleotides are essentially the same as nucleotides except that they contain a hydrogen group on the 3 carbon instead of a hydroxyl group (OH) (Fig. 2.13). In Sanger sequencing, many copies of a DNA strand that needs to be sequenced are replicated using DNA polymerase in the presence of normal nucleotides as well as the appropriate proportion of dideoxynucleotide bases. The enzyme starts replicating from 5' to 3' end, adding first a C (correspond to the first G at 5' end of the template strand) or ddC (dideoxynucleotide C). If a ddC is incorporated then this will prevent further addition of the nucleotides as a phosphodiester bond cannot form between the dideoxynucleotide and the next incoming nucleotide, and thus the DNA chain is terminated. If a normal base C is incorporated as the first base then more nucleotides can be added further. Finally the DNA product is separated using gel electrophoresis. In gel electrophoresis, the short fragments travel furthest. In Fig. 2.13, C is the first base in the complementary strand. The next base is again a C, then G and so forth. In this way, the entire complementary nucleotide sequence can be read. Sanger sequencing greatly simplified the DNA sequencing and was commonly used for almost two decades.

Sanger and Maxam–Gilbert sequencing were performed manually and was labor–intensive. In 1986, Leroy Hood et al. published an automated method to perform Sanger sequencing that used

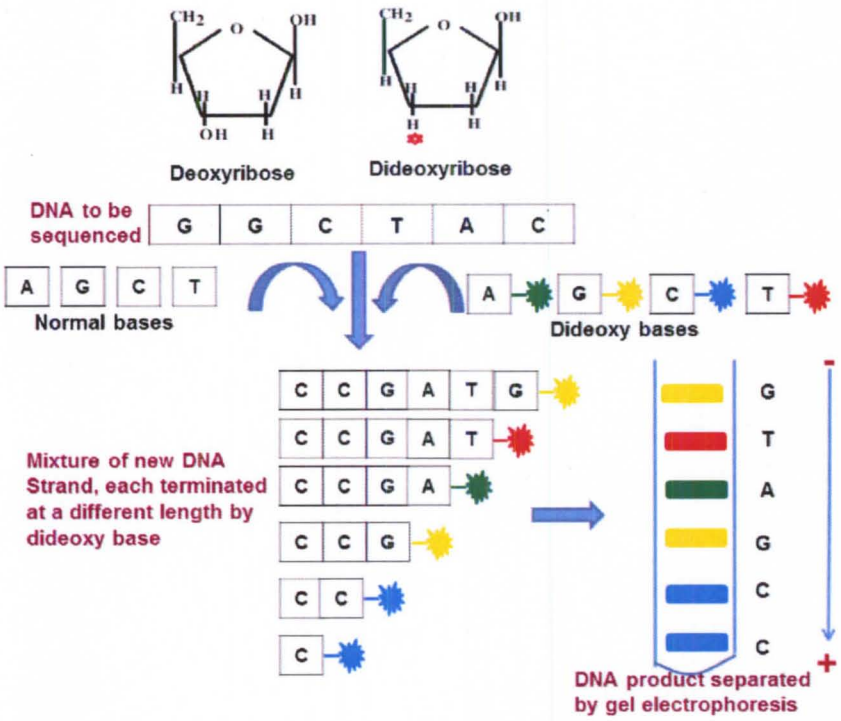

Figure 2.13: Sanger method of DNA sequencing.

fluorescently labeled dideoxynucleosides [26]. This sequencing was automated by machines where fluorescence was detected by laser.

Next–generation sequencing

Up until mid 2000s machines based on Sanger sequencing method were used for sequencing. The commercialization of genome sequencing started in 2004, when Roche (454) came up with first massive parallel pyro–sequencing technique with the ability to sequence virtually any genome at a cost effective price. This method was based on a "sequencing–by–synthesis" method that relies on the detection of pyrophosphate released during nucleotide incorporation. This method allows sequencing of a single strand of DNA by synthesizing the complementary strand along it, one base pair at a time, and detecting which base was actually added at each step. The amount of light produced is proportional to the number of nucleotides incorporated. One limitation of this technique is the inability to distinguish long homopolymer runs in the sequence [27, 28]. Illumina also uses a "sequencing–by–synthesis" method using a proprietary reversible terminator–based method that

enables detection of single bases as they are incorporated into growing DNA strands. Since all four reversible terminator-bound dNTPs are present during each sequencing cycle, natural competition minimizes incorporation bias. Helicos Biosciences sequences single molecules of DNA or RNA using a "sequencing–by–synthesis" approach. Applied Biosystems (ABI) uses a "ligation–based sequencing" protocol. It uses DNA ligase to amplify fragments. Multiple cycles of ligation, detection and cleavage are performed with the number of cycles determining the eventual read length [29].

Next–next generation sequencing

The Year 2010 was another landmark in the DNA sequencing. Pacific Biosciences came up with two proprietary technologies: Single Molecule Real Time Sequencing (SMRT sequencing) and fluorescently labeled phospholinked nucleotides. Using these two technologies and a Zero Mode Waveguide (ZMW) nanostructure arrays, sequencing can be done in real–time [30]. This technology produces longer reads but has a relatively high error rate. The Personal Genome Machine(PGM) by Ion Torrent works on the concept that each natural incorporation of DNA by a polymerase result in the release of hydrogen ion (H+) which changes the pH of the solution. By measuring the pH it can be determined whether a nucleotide is incorporated [31, 32].

A comparison of sequencing platforms is shown in Table 2.1.

Table 2.1: Comparison of sequencing platforms

Platform	**Read length (bases)**	**bases per run (Gigabases)**	**Run time**	**NGS chemistry**
Roche/454 (GS FLX titanium XLR70)	450	0.45	10 hrs	Pyrosequencing
Applied Biosystems (SOLiD 5500xl)	60	20 – 25 Gb/day	7 days	Sequencing by ligation
Illumina/Solexa (GA IIx)	35-150	11 - 57	2 – 14 days	Sequencing by synthesis / Reversible terminator
Helicos (Heliscope)	35 avg.	28	8 days	Sequencing by synthesis / tSMS
Pacific Biosciences (PacBio RS)	2200	–	10 hrs	SMRT
IonTorrent (PGM)	200 bp max	10	2 hrs	pH difference, semiconductor chip

NGS application

NGS technologies have a wide range of applications, and more are being discovered. It has been used successfully in applications such as variant discovery, targeted resequencing, *de novo* assembly of bacterial genomes, sequencing personal genomes and possible usage in personalized medicine, cancer diagnosis, genes, transcripts and proteins discovery and many other areas.

2.4 Genome alignment and assembly

Next-generation sequencing (NGS) and next–next generation sequencing techniques are parallelized high–throughput methods that can produce millions of short sequences (reads) in a very short period of time. The read lengths varies for different platforms ranging between 40-500 bp for NGS and higher for next–NGS. One of the crucial steps of NGS analysis is to map these reads back to their sequence of origin. These reads can be aligned to either a reference sequence or can be assembled *de novo*. Reference–based assembly is easier and often performed; however, in some cases it is unable to perform mapping accurately. For example, a read may belong to repetitive regions or the read is not present in the reference genome at all. This section explain two reference based assembly algorithms which is used in this work for aligning genomic sequences to a genome. BLAST is also discussed here as it is basis for another BLAT (discussed next).

- *BLAST (Basic Local Alignment Search Tool)* is a heuristic algorithm for computing optimal "local alignments" between a query sequence (Q) and a database (D) containing one or more subject sequences. BLAST has two main components; the first component implements a search algorithm for finding local alignments and the second component uses an associated theory for estimating the statistical significance of solutions to help distinguish true similarities from ones that are due to chance. A BLAST search begins by indexing all words of length k from the query and then matching each of these words against database sequences. For nucleotide–to–nucleotide searches, each of these matches must be exact whereas for protein–to–protein searches the matching must have a similarity score $\geq T$ i.e. threshold. These scores are

determined using a substitution matrix such as PAM or BLOSUM. When a word match is found, BLAST attempts to extend the alignment in both directions. BLAST continues this extension in search of a *high-scoring segment pair (HSP)*. An HSP cannot be extended further to the left or right if the score drops significantly below the best score achieved on part of the HSP [33]. The alignments found by BLAST during a search are scored, and assigned a statistical value, called the "Expect Value". The "Expect Value" is the number of times that an alignment as good or better than that found by BLAST would be expected to occur by chance, given the size and composition of both the database and query. BLAST's default value '10' ensures that no biologically significant alignment is missed; however, high quality alignments can be obtained by lowering this value.

BLAT: The BLAST–Like Alignment Tool

W. James Kent, in 2002, developed BLAT (BLAST–Like Alignment Tool) tailored to highly similar sequences, which was faster (500 times faster mRNA/DNA alignment and 50 times faster for protein sequences) than BLAST. BLAT is similar to BLAST in the way that both find HSPs. In case of DNA, BLAT works by keeping an index of the entire genome in memory as the target database. The index uses less than a gigabyte of RAM for the human genome and consists of all non-overlapping 11-mers except for those heavily involved in repeats. DNA BLAT is designed to quickly find sequences of $\geq 95\%$ similarity of length 40 bases or more. However, it may miss more divergent or short sequence alignments [34]. For proteins, BLAT uses 4-mers and finds protein sequences of $\geq 80\%$ similarity to the query of length ≥ 20 amino acids. The basic difference between BLAST and BLAT is that the former indexes the query sequence while latter indexes the database sequence. BLAST triggers an extension when one or two hits occur, while BLAT can trigger extensions on any given number of perfect or near perfect matches. BLAST returns each area of homology as separate alignments, while BLAT stitches them together into larger alignments (`http://genome.ucsc.edu/FAQ/FAQblat.html`).

BLAT uses a seed-and-extend approach for alignments. In the seed (search) stage, it uses an index to find regions in the genome that are possibly homologous to the query sequence. In the extend (alignment) stage, it perform an alignment between such regions and then stitches together the aligned regions (often exons) into larger alignments (typically genes). BLAT provides three different searches in the seed stage [34]: 1. Searching with single perfect matches, 2. Searching with single near perfect matches, and 3. Searching with multiple perfect matches. The following text describes the first of the three searching option provided by BLAT.

Searching with single perfect K-mer matches:

K: The K–mer size

M: Match ratio between homologous areas, $\sim$ 98% for cDNA/genomic alignments within the same species, $\sim$ 89% for protein alignments between human and mouse.

H: The size of a homologous area. Generally 50 – 200 bp. for human exon

G: Database size, e.g. 3 Gb for human.

Q: Query size.

A: Alphabet size, 20 for amino acids, 4 for nucleotides.

Assuming that each letter is independent of the previous, the probability that a specific K–mer in a homologous region of the database matches perfectly the corresponding K-mer in the query is:

$$p_1 = M^K$$

Let $T = \lfloor \frac{H}{K} \rfloor$ denote the number of non–overlapping K–mers in a homologous region of length H.

Sensitivity: The probability (of a hit) that at least one non–overlapping K–mer in the homologous region matches perfectly with the corresponding K–mer in the query is:

$$P = 1 - (1 - p_1)^T = 1 - (1 - M^K)^T$$

Specificity: The number of non-overlapping K–mers that are expected to match by chance, assuming all letters are equally likely, is:

$$F = (Q - K + 1) * (G/K) * (1/A)^K$$

These formulas can be used to predict the sensitivity and specificity of single perfect nucleotide K–mer matches as a seed-search criterion. It was shown that for EST alignments of nucleotide sequences, a value of K = 14 or less gives at least 99% of the sequences that have 5% or less sequencing noise.

The extend stage performs a detailed alignment between the query sequence and the homologous regions returned by the previous stage. If a K-mer in the query hits multiple K-mers in the homologous region, the K-mer is extended by one repeatedly until the map is unique or the K-mer exceeds a certain size. These hits are then extended as far as possible allowing no mismatches, and the overlapping hits are merged. These extended hits that follow each other in the query and target sequences are linked together to get the alignments. In some cases, stitching of the alignments may be performed when a gene is scattered across multiple homologous regions.

Bowtie

Bowtie is an ultrafast and memory efficient short–read aligner for aligning DNA sequences to large genomes. The Bowtie indexer can compress and index the whole human genome into 2.3 GB of memory. It can align 25 million, 35-bp reads onto the human genome in an hour with a peak memory footprint of 1.3 GB. Bowtie can align reads ranging from 4 bases to 1,024 bases. It uses the Burrow–Wheeler (BW) algorithm with Ferragina–Manzini (FM) index to find the exact match. To allow mismatches and favor high quality reads, it extends the algorithm by

using a quality–aware backtracking algorithm. It also uses 'double indexing' to limit excessive backtracking while performing inexact alignments [35, 36]. A Burrows–Wheeler transform (BWT) of a text T is constructed as in Figure 2.14. Initially a \$ or any special character (lexicographically smaller than all the possible characters) is appended to the input sequence. Next, all cyclic rotations of this text are found in the matrix and are sorted lexicographically. The last column of each row in the sorted matrix form the actual transform $BWT(T)$ and is of the same length as the text T. The remarkable property of $BWT(T)$ is reversibility, allowing the original text to be recreated.

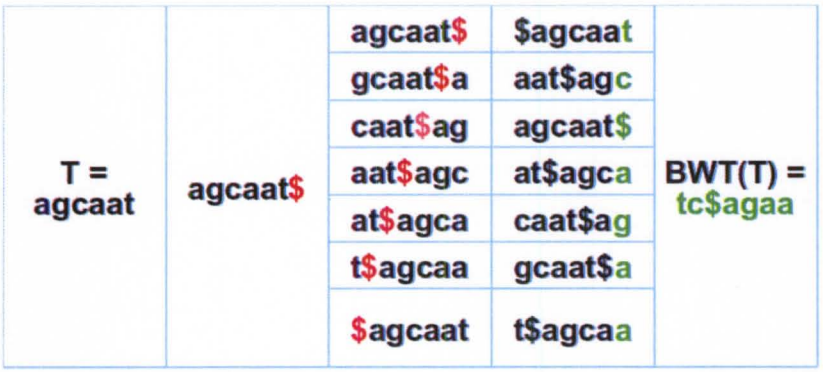

Figure 2.14: Constructing Burrows–Wheeler transform.

The exact match alignment in Bowtie uses the above sorted matrix and calculates the range of matrix rows beginning with successively longer suffixes of the query. Bowtie also addresses inexact alignments that may occur due to sequencing errors or polymorphisms. The algorithm is similar to that of exact match, calculating matrix ranges for successively longer query suffixes. At any point when the matrix range becomes empty, Bowtie may select an already matched query position and substitute with a different base. This introduces a mismatch into the alignment and proceeds with finding the matrix range again. Each substitution is consistent with the alignment policy. Bowtie is a greedy approach and in the case where multiple substitution positions are found, the algorithm selects the position having the minimum quality value. Bowtie avoids excessive backtracking while balancing the sensitivity of the aligner

by maintaining two indexes: a forward index and a mirror index. Bowtie limits the maximum number of backtracks to 125.

Bowtie2 is an upgraded version of Bowtie for aligning comparatively long sequencing reads of about 50 bp up to 1000 or more. Bowtie2 supports gapped, local and paired-end alignment modes. For sequences shorter that 50 bp, Bowtie sometimes performs better than Bowtie2.

Other software are available for performing reference–based assembly. This include: MAQ (Mapping and Assembly with Quality) [37] is based on the mapping quality concept, ELAND (Efficient Local Alignment of Nucleotide Data) [38] which searches DNA files for short DNA reads allowing up to two errors per match and SOAP (Short Oligonucleotide Alignment Program) [39, 40] which uses a Burrows–Wheeler algorithm to perform alignment and are fast and memory–efficient.

De novo assembly algorithms assemble the short reads to create full–length sequences. These types of assemblers are complex, time consuming and memory inefficient as they require many more comparisons (in the worst case, all possible comparisons) to construct a sequence. Examples of such assemblers include Velvet [41], ALLPATH-LG [42], Quality Value Guided SRA (QSRA) [43] and VCAKE [44]. These algorithms are outside the scope of this thesis work and thus not explained.

2.5 The Human Genome Project

The Human Genome Project (HGP) started in October 1990, initially estimated to sequence the whole human genome in about fifteen years at $200 million per year at a cost rate $1 per base pair. Sponsored by US Department of Energy (DOE) and National Institute of Health (NIH), the specific goal of HGP was to identify all the genes in human DNA, determine the sequences of the three billion chemical base pairs that make up human DNA, store this information in databases, improve tools for data analysis, transfer related technologies to the private sector, and address the ethical, legal, and social issues (ELSI) that may arise from the project. The advent of PCR technology by *Kary Mullis* [45] and other sequencing methods such as the whole genome shotgun (WGS) sequencing, cDNA technology and others fueled the competition. Private players including Celera started using WGS to rapidly sequence the genome. Celera, led by J. C. Venter sequenced

the whole genome of *Haemophilus influenza* in 1995 with this brute–force shotgun strategy [8]. The competition between public and private was so high that the completion of 'draft' genome was announced on June 26^{th}, 2000. *Science* and *Nature* published the genome paper in the same week of February 2001 [46] [15]. The first draft of the human genome contained roughly three billion base pairs and was almost 90 percent complete. A startling finding of this first draft was that the number of human genes appeared to be significantly fewer than previous estimates, which originally ranged from 50,000 genes to as many as 140,000. The full sequence was completed and published in April 2003.

2.6 Expressed sequence tags (ESTs)

An expressed sequence tag (EST) is a short (200 to 800 base pair in length), unedited single–read sequence generated by sequencing cDNA. The cDNA itself is prepared from mRNA by an enzyme called *reverse transcriptase* [47]. Once the cDNA representing an expressed gene has been isolated, a few hundred nucleotides can be sequenced from either end to create 5'ESTs or 3'ESTs. ESTs have been primarily used in the discovery of novel human genes and genomic coding regions since they represent transcribed sequence. ESTs are a rapid and inexpensive method for understanding an organism's transcriptome that may be helpful in the prediction of their protein products and ultimately their function. ESTs of length 150 to 400 base pairs have been shown to contain sufficient information for similarity searching and mapping which permit the design of precise probes for DNA microarrays that then can be used to determine the gene expression and other downstream exploratory analyses.

2.7 Microarrays

A DNA microarray is a collection of microscopic DNA spots attached to a solid surface such as glass or silicon chip. Each DNA spot contains picomoles of a specific DNA sequence or oligonucleotides, known as probes. A microarray chip may contain tens of thousands of spots and each of these spots may contain millions of oligos or DNAs of a particular gene for that spot. A microarray works

by exploiting the ability of an mRNA molecule to bind, or hybridize to, the DNA template from which it originated. By using an array containing many DNA samples, scientists can determine, in a single experiment, the expression levels of hundreds or thousands of genes within a cell by measuring the amount of mRNA bound to each site on the array. A microarray may be used mostly in three different ways: 1) In *Microarray Expression Analysis*, expression of a set of genes in one particular condition can be compared to the expressions of another set of genes in another condition, 2) *Microarray Mutation Analysis* is used for performing SNP detection and 3) *Comparative Genomic Hybridization* is used mostly by Agilent Technologies to assess genome content in different cells or closely related organisms.

Three types of microarrays are widely used for analysis of gene expression. The first is based on short oligonucleotides (oligos), the second is based on long oligos and the last is based on cDNA technology. Though the short oligo (25–30 base pairs) arrays are the mainstay for expression analysis, long oligo (50–80 base pairs) arrays are gradually gaining popularity. The cDNA arrays are variable in length and are also popular among scientists because of flexibility in array synthesis that it gives to the user.

The major steps while performing microarray experiments are as follows: [48]:

- *Sample Preparation and Labeling:* The RNAs from the tissue of interest are extracted and are reverse transcribed to produce cDNAs. These cDNAs are then labeled depending on the platforms being used. Affymetrix uses a single channel biotin–labeled complimentary RNA for hybridization. Other cDNA arrays use a dual channel approach to label the samples (e.g. control labeled with green dye and the contrasting sample labeled using red dye).

- *Hybridization:* These cDNAs are allowed to hybridize onto the same glass slide. A cDNA sequence will hybridize to specific spots that contain its complimentary sequence. Hybridization is a complex process and highly dependent on factors such as temperature, humidity, salt concentration volume of target solution etc, and may be performed either manually or by robots.

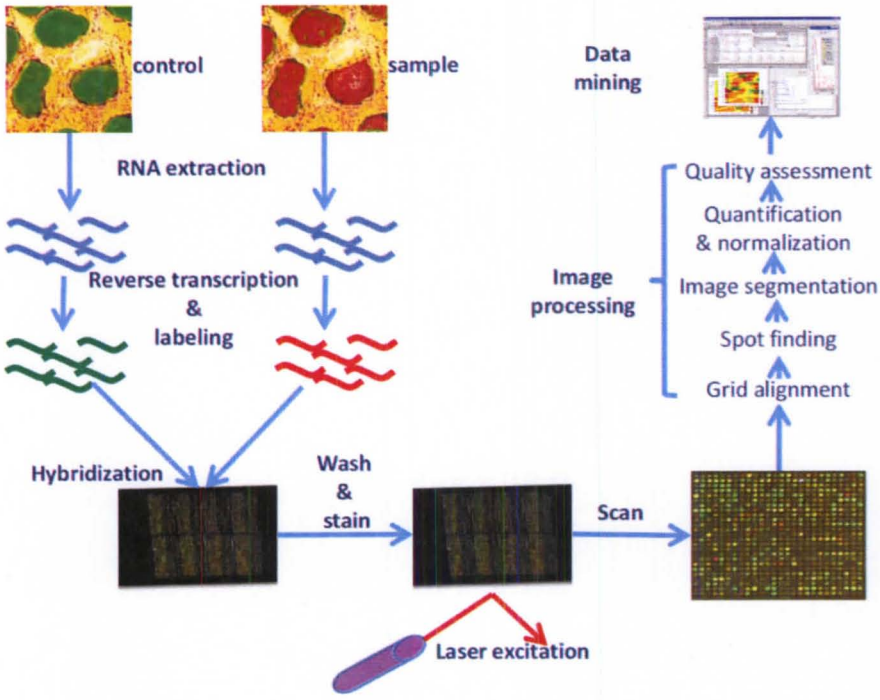

Figure 2.15: Microarray analysis steps [49]

- *Washing:* Washing is performed to remove extra hybridization solution to ensure that only the labeled target on the array is the actual target of interest.

- *Image Acquisition:* After hybridization, the dyes are excited by a laser at an appropriate wavelength and scanned by laser that reads the surface. The fluorescence detected are stored as a digital image usually in tagged image file format (.tiff) into the computer [48].

- *Image Processing:* In this step, potential spots are found and distinguished from spurious signals. These spots are then quantified by combining the pixel intensity values into unique quantitative measures that can be used to represent the expression level of the gene deposited in a particular spot. This amount is directly proportional to the mRNA present in the solution that hybridized the chip.

- *Data transformation and normalization:* The signal intensities are usually transformed and normalized in several steps in order to improve comparability and signal/noise ratio. The transformation step may include subtraction of an estimated background signal and logarithmic

transform or subtraction of the reference signal. Microarray experiments involve many steps and each step can introduce variabilities in the results. These variabilities can be minimized by performing normalization on the data, including total intensity normalization, quantile normalization, lowess normalization, linear regression and Chen ratio statistics [50].

- *Analysis of Gene Expression Data*: Once the normalized data is available, various techniques may be used to determine subsets of genes that are significantly changed between conditions. Determination of the sets of differentially expressed genes is a statistical problem that involves calculation of a p–value for significance. Example of different methodologies that may be used include *Significant Analysis of Microarray(SAM)* [51], *Random Forest* [52], *entropy based gene selection method* [53] and *False Discovery Rate(FDR)* [54]. A number of software tools are available to find differentially expressed genes in microarray including SAM, Limma [55], Multtest [56], twilight [57], Nudge [58], penalizedSVM [59] and RandomForest.

- Once an important gene set is derived from the steps above, a scientist may apply different algorithms to accomplish their tasks, such as classification, clustering and phylogenetic analysis [60] [48].

Efforts have been taken to standardize microarray data. The Microarray Gene Expression Data (MGED) society has proposed MIAME (Minimum Information About a Microarray Experiment) standard that requires the submitter of the data to furnish some required information such as raw data for each hybridization, normalized data and sample annotation data processing protocol etc. This will reduce ambiguity in the data and lead to better interpretation, verification and reusability of the microarray data. Public repositories such as ArrayExpress at EBI, GEO at NCBI and CIBEX at DDBJ are designed to accept MIAME compliant data. In addition, most journals (complete list can be found on `http://www.mged.org/Workgroups/MIAME/journals.html`) now require MIAME compliant data for publishing a microarray based paper [61].

While microarray techniques have some inherent limitations, they are useful in helping scientists determine differentially expressed genes, pathway analysis of genes, drug development and drug response, therapy development, tumor classification and clustering, tracking disease progression,

alternative splice detection, phylogenetic analysis, mapping deleted or duplicated regions in genome and mapping genes to phenotype [62] [63].

2.8 Genome annotation

Genome sequencing projects produce huge amount of sequencing data. *Genome annotation* is the process of adding the layers of analysis and interpretation to these sequences necessary to extract their biological significance and place these into the context of our understanding of biological processes [64]. Genome annotations can be broadly categorized into three levels:

- *Nucleotide* or *structural annotation* has the goal of determining the location of sequences and where do they found on genome including the start and end locations, ORF locations, locations of non–coding RNAs and regulatory regions, exon landmarks, repetitive regions and mapping variations.
- *Functional* annotations are more concerned with what these sequences do, what are the corresponding proteins and their putative biological and biochemical functions.
- *Process* annotations relate these sequences to various processes such as cell cycle, cell death, embryogenesis, metabolism etc. and how do they behave in a system (regulations, interactions).

Nucleotide annotation

The first step in genome annotation is to identify the location of genetic elements such as genes, genetic markers, tRNAs, rRNAs, ncRNAs, repeat regions and ORFs., and the next step is to attach biological information to these elements. There are a number of algorithms that automatically annotate these entities.

Gene prediction software identifies the regions of genomic DNA that encode genes. This includes protein-coding genes as well as RNA genes, but may also include prediction of other functional elements such as regulatory regions. In *ab initio* gene finding, the DNA sequence is systematically searched for certain signals or sequences that indicate the presence of gene. Examples include GENSCAN [65] and geneid [66] algorithms. Advanced gene finders such as GLIMMER (Gene Locator

and Interpolated Markov ModelER) [67] and GeneMark [68] use complex probabilistic models, such as hidden Markov models, in order to combine information from a variety of different signal and content measurements. Other algorithms such as mSplicer [69], CONTRAST [70] and mGene [71] use machine learning techniques like support vector machines for successful gene prediction. Database projects such as RefSeq, Entrez Gene, Ensembl and ENCODE are involved in annotation of genes and will be described shortly. Similar to gene prediction algorithms, there are algorithms that search for non–coding RNAs (such as tRNA, rRNA and snRNA) and transcriptional regulatory regions. tRNAscan–SE [72] detects tRNA, RNAmmer [73] uses HMMER to annotate rRNA, RNAmicro [74] and miRNAminer [75] recognize microRNA. Annotations for transcription binding sites are available in curated databases such as TRANSFAC [76] and PROSITE [77]. RepeatMasker screens DNA sequences in FASTA format against a library of repetitive elements and returns a masked query sequence along with the annotated masked regions. There are a number of algorithms available to perform SNP detection and segmental duplication detection. NCBI's dbSNP is a comprehensive database of SNP annotation.

Functional annotation

Functional annotations seeks to compile a definitive catalog of the functions of specific genomic regions of the organisms such as protein naming and putative functions. Putative functions can be computationally assigned using sequence similarity with algorithms such as BLASTP or PSI-BLAST against the curated database of proteins. UniProtKB/Swiss-Prot [78] is based on this methodology. The Pfam (Protein family) [79] database is a large collection of protein families and use probabilistic hidden Markov models (HMMs) for annotating proteins based on functional motifs. NCBI maintains a protein database which is a collection of sequences from several sources, including translations from annotated coding regions in GenBank, RefSeq and third party annotation, as well as records from SwissProt, PIR [80], PRF, and PDB (Protein Data Bank) [81].

Process annotations

Process annotations annotates sequences to its biological processes. For instance, the Gene Ontology (GO) project is a collaborative effort of associating process level information to the genetic products. The specific aims of the project is: 1) the development and maintenance of the ontologies themselves; 2) the annotation of gene products; and 3) development of tools that facilitate the creation, maintenance and use of ontologies. The GO ontology covers three domains: molecular function, biological process and cellular component. Molecular function describes activities, such as cell cycle, cell death and embryogenesis, that occur at the molecular level. A biological process is used for broader biological goals, such as meiosis. A cellular component is just that, a component of a cell, but with the provision that it is part of some larger object [82].

2.9 Annotation databases

A large number of annotation databases are available that annotate genomes or sequences produced by various high–throughput methods. The Nucleic Acid Research (NAR) 2012 database issue [2] features 1380 databases covering various aspects of molecular biology including sequences, annotations, gene expression, structures, pathways and diseases. This section gives a brief introduction of some of the popular annotations databases available.

Ensembl

The Ensembl project, developed jointly by the EBI and the Wellcome Trust Sanger Institute, has been used for the annotation, analysis and display of vertebrate genomes [83]. Since its inception in 2000, Ensembl added support for many more organisms in its database. Ensemble uses genebuild pipeline to automatically annotate the protein coding genes, pseudo–genes, non-coding RNAs and EST–based genes. Ensembl provide genome specific sequence data for all the ensembl transcripts and genes in different format through its *ftp* website `ftp://ftp.ensembl.org/pub/current_fasta/`. Unspliced gene sequences, unspliced transcript sequences, exon sequences, cDNA sequences, flanking region sequences and many more can also be downloaded.

HUGO Gene Nomenclature Committee (HGNC)

HGNC [84] assigns nomenclature to the human genes following well defined guidelines and store these into its database. As of May 31^{st}, 2012, it has approved almost 33,000 symbols; a vast majority of these are protein–coding genes ($\approx 19,000$), but also include symbols of pseudogenes, ncRNAs, phenotypes and genomic features. HGNC also interact with other organism specific nomenclature committees on regular basis.

The International Nucleotide Sequence Database Collaboration (INSDC)

The INSDC is a collaborative step to maintain a comprehensive database of nucleotide sequences. It comprises of DNA Data Bank of Japan (DDBJ), the European Molecular Biology Laboratory (EMBL) and GenBank at NCBI which exchange their data on a daily basis to achieve maximal synchronization.

GenBank

NCBI's GenBank [85] is an annotated genetic sequence database of publicly available DNA sequences and their protein translation. As of April 2011, there are approximately 126,551,501,141 bases in 135,440,924 sequence records in the traditional GenBank divisions and 191,401,393,188 bases in 62,715,288 sequence records in the WGS division `http://www.ncbi.nlm.nih.gov/genbank/`. Sequence data can be submitted through NCBI's GenBank submission system program such as *BankIt* and *Sequin*.

RefSeq

NCBI's RefSeq [86] is a curated, annotated and non–redundant collection of DNA, RNA and protein sequences. Sequences from plasmids, organelles, viruses, archea, bacteria, and eukaryotes are included in the database. This database can be searched using genomic location, sequence , or text as query. It is based on records submitted to the INSDC. RefSeq has support for genome annotation, gene characterization, comparative genomics, reporting sequence variation, and expression studies.

RefSeqGene

NCBI's RefSeqGene project is a subset of RefSeq and defines genomic sequences to be used as reference standard for well characterized genes. It provides more stable gene–specific genomic sequence for each gene along with upstream and downstream flanking regions.

UniProt

UniProt [87] provides a comprehensive, high quality and freely accessible resource for protein sequences and their functional annotation. It consists of two sections: *Swiss-Prot* where the annotations are performed manually and reviewed, and *TrEMBL*, where the annotations are performed automatically and are not reviewed.

Entrez

The NCBI's Entrez [88] is a powerful database to search and retrieve sequences, structures and references for a particular entity. It also provide views of genes, proteins and chromosome maps. Using a single query, several linked databases can be searched including ESTs, Gene, Genome, GEO dataset, GEO profiles, probe, PubMed, SNP, structure, taxonomy, UniGene and UniSTS.

Gene

The NCBI's Gene database supplies gene specific information including nomenclature, Reference Sequences (RefSeqs), maps, pathways, variations, phenotypes, and links to genome-, phenotype-, and locus-specific resources worldwide.

dbEST

The dbEST database, a division of GenBank stores sequence data and annotation information for cDNA sequences or ESTs for a number of organisms. dbEST provide a robust sequence resource that can be exploited for rapid gene discovery, genome annotation and comparative genomics, guiding

SNPs (Single Nucleotide Polymorphism) discovery, gene structure prediction, investigating alternative splicing and discovering cancer biomarkers [89] [90] [91]. Scientists and researchers across the world and genome sequencing centers submit tens of thousands of ESTs everyday to NCBI's GenBank. As of May 1, 2012, the total number public entries of ESTs in NCBI's dbEST repository was 72,693,656 across more than 2000 organisms `http://www.ncbi.nlm.nih.gov/dbEST/dbEST_summary.html`.

The UCSC Genome browser

The UCSC Genome Browser [92] is an online interactive website to access genome annotation data for a large number of organisms in a variety of ways. It enables researchers to visualize and browse entire genomes on annotation track for different types of data including gene locations, SNPs, proteins, expression, comparative analysis, homology etc. A user can also define and view his own custom track. This is an open source project and all its data is freely available to download via `ftp` for non–commercial use. Different utilities and softwares such as BLAT, liftOver and The Genome Browser can be downloaded freely. The Genome Browser also hosts *proteome browser* and browsers for microbial genomes.

GEO

NCBI's Gene Expression Omnibus (GEO) [93, 94] is a public functional genomics data repository that archives and distributes data from high–throughput experiments such as microarrays and next–generation sequencing, serial analysis of gene expression, protein arrays and ChIP–chip data. The contents in GEO can be describes as platforms, series, samples and datasets. The contents of GEO can be browsed or text queried. As of May 15, 2012, GEO contains 10,081 platforms, 741,557 samples, 30,107 series and 2,720 datasets.

ArrayExpress

ArrayExpress from EMBL–EBI is a database functional genomics experiments where data can be queried or downloaded using MIAME (Minimum Information About a Microarray Experiment) [61] standards. It can be queried using accession or keywords.

Affymetrix® NetAffx™

Affymetrix® GeneChip® is one of the microarray platforms that is used widely and most popular among scientists and researchers. In this technology each gene is typically represented by a set of 11–20 pairs of probes. Gene expression is measured by extracting mRNA from the cells or tissues of interest and hybridizing the mRNA sample to the 25–mer probes on the microarray (Fig. 2.16). Each expressed transcript is represented on an array by a series of probe pairs known as a probe set. Each pair consists of a perfect match probe, with its 25–base sequence identical to the gene of interest, and a mismatch probe, whose sequence is the same as the perfect match except for position thirteen, where the base is set to the complementary of the perfect match. Each probe set on the Affymetrix® arrays consists of 11 probe pairs, and is given a unique identifier consisting of a seven digit number, followed by the optional characters _s, _a, or _x, and ending in _at [`http://www.affymetrix.com/support/technical/index.affx`]. Affymetrix® probe sequences can be downloaded from the NetAffx™ Analysis center at Affymetrix® website. Affymetrix® probes give excellent coverage of known genes. For the human genome, as of January 4, 2007, 24,198 of the 24,259 (99.7 percent) sequences present in the RefSeq database were covered by four or more probes on the Affymetrix® exon array. More than 98 percent of RefSeq and more than 90 percent of the Ensembl protein-coding transcripts were covered by 10 or more probes [96].

The NetAffx™ [97] analysis center details and annotates probesets on Affymetrix®'s GeneChip® arrays. It annotates each probeset with its composition: the probes that constitute the probeset, sequence information and the genomic locations, protein sequence–level annotations and associated ontological terms. Each probeset is structurally annotated using GenBank, LocusLink and Swiss–

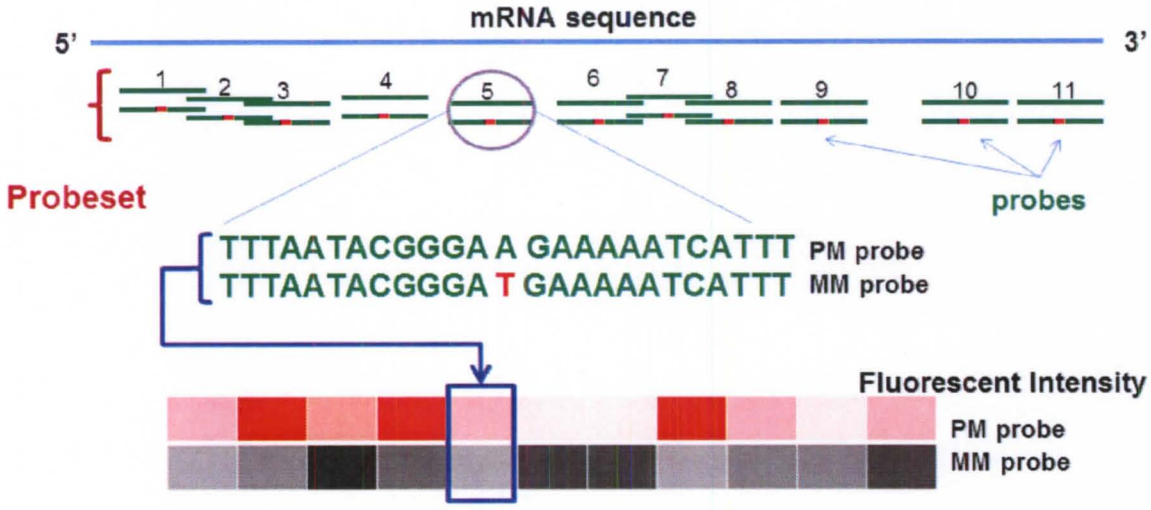

Figure 2.16: Affymetrix GeneChip® design [95]

Prot identifiers as well as functional information in terms of GO terms and GenMAPP pathways. It is a searchable database and can be queried using annotation terms and probeset IDs.

2.9.1 Agilent Technologies's eArray utilities

Agilent Technologies manufactures a variety of catalog and custom long-oligonucleotide (60-mer) microarrays that can be used in multiple two-color microarray applications. Optimized methods and techniques have been developed for two such applications: gene expression profiling and comparative genomic hybridization. Methods for a third technique, location analysis, are evolving rapidly. A key component of Agilent's Custom Microarray Design process includes the array layout and basic QC components of the design process. In array layout which is an aspect of the collaborative design service that gives you complete flexibility in your design. The processes enables you to rapidly iterate and print new array layouts. The user has the flexibility to design probes of size ranging from 25-60. You can also randomize probe placement on the microarray. Agilent microarrays are used in a number of different applications such as gene expression profiling, microarrays, comparative genomic hybridization (CGH) and ChIP on Chip. Agilent provides a web portal in the form of *eArray* as a mean to create custom microarrays, enrichment libraries, and mutagenic oligos online. eArray also

provides facilities including download of the latest annotations for each probe and compare groups of probes.

CHAPTER 3

INTERVAL–TREES FOR REPRESENTATION OF OVERLAPPING GENETIC ENTITIES

3.1 Introduction

The origin of all nucleic acid and protein–based entities is genomic DNA sequences. For species where a reference genome is available, these DNA sequences can be aligned to the reference genome and assigned absolute numeric coordinates on the genome. These coordinates consist of information such as the start and end location(s) on the genome, underlying gaps, and intron–exon boundaries. Their lengths range from one base (SNP) to kilobases (gene locus), or even megabases (chromosomal bands). Different databases annotate these entities differently and their annotations tend to show a large degree of overlap. Fig. 3.1 shows the extent of overlap between the intervals of different annotations in the region of human BRCA2 (Breast Cancer 2, early onset) gene on chromosome 13.

3.2 Interval representation of genetic entities

Having been assigned numerical coordinates, possibly with gaps (intronic regions), a GE can be represented as an interval on a genomic scale. As shown in Fig. 3.1, a genetic entity may be a continuous region (microarray probe, SNP etc.) on the genome or may contain gaps in between (genes and transcripts). Availability of different types of GEs in different databases, each with different size (different granularity), complicates its representation as a large number of these entities overlap each other by sharing same region on the genome.

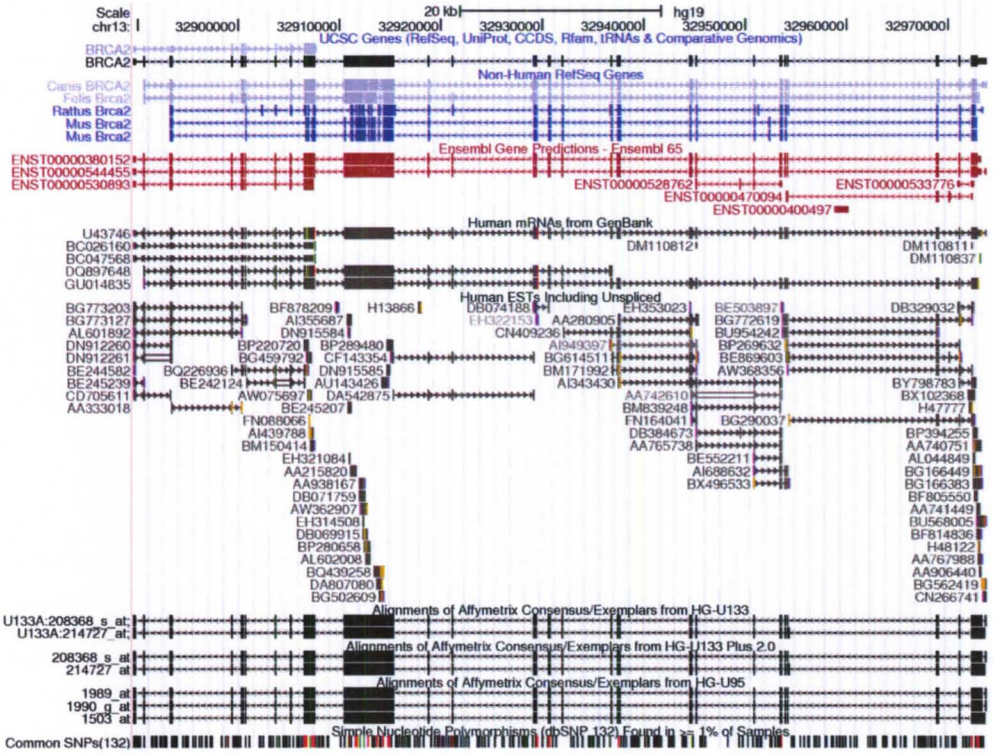

Figure 3.1: Different types of GEs in the region of the human BRCA2 gene. Generated on the UCSC genome browser [98].

An interval is a convenient representation of events spanning a continuous period of time or space. Time examples include transactions in a bank, or time spent on a web page whereas space intervals include map features, photographs, words in a document, or genetic entities (GE). Interval structures can also be found in application areas such as life sciences, computer graphics, databases, robotics, computational geometry and geographic information systems. All of these problems have a similar structure, where one entity shares space or time with many other entities.

An interval is an ordered pair $[t_1, t_2]$ of real numbers with t_1, the low– and t_2, the high–end point. If i is an object with an associated interval, then i can be represented as $[t_1, t_2]$, with $t_1 \leq t_2$. An interval may be closed, open or half open. A closed interval can be represented as:

$$i = [t_1, t_2] : \{t \in \mathbb{R} : t_1 \leq t \leq t_2\}$$

Open intervals are represented as:

$$i = (t_1, t_2) : \{t \in \mathbb{R} : t_1 < t < t_2\}$$

Fig. 3.2 shows different examples of one- and two-dimensional interval structures. Fig. 3.2(a) shows overlapping intervals in one-dimension. An interval i is shown with low end point as t_1 and high end point as t_2. Fig. 3.2(b) and Fig. 3.2(c) show the overlapping intervals in 2-dimensions.

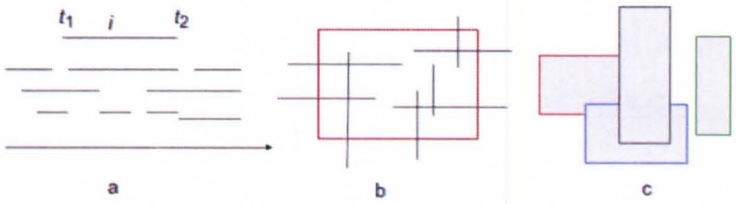

Figure 3.2: Overlapping intervals in one (a) and two (b,c) dimensions.

James F. Allen, in his paper "Maintaining knowledge about temporal intervals" [99], elaborated thirteen basic relations among temporal (time) intervals. These intervals are distinct (no pair of definite intervals can be related by more than one of these relations) and exhaustive (any pair of time intervals can be represented by one of these relations) and are shown in Figure 3.3. Each relation relates two temporal intervals X and Y, with the time moving from left to right. These relations are sorted by the degree to which X begins before Y and then within that by the degree to which

X ends before Y. All the thirteen basic relations are constituted by six relations and their inverses and equality relation. These are shown in Table 3.1. $X < Y$ (X precedes Y) means the interval X completed before Y started. The inverse relation for this is $>$ (preceded by). Whenever $X < Y$ is true, its inverse $Y > X$ (Y preceded by X) is always true.

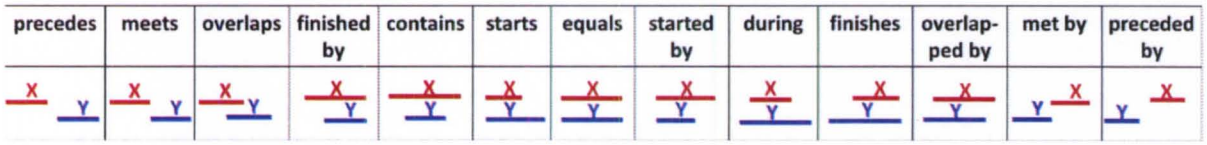

Figure 3.3: The thirteen interval relations defined by James F. Allen.

Table 3.1: Basic temporal relations and inverses.

Relation	**Symbol**	**Inverse relation**	**Symbol**
precedes	$<$	preceded by	$>$
meets	m	met by	mi
overlaps	o	overlapped by	oi
finished by	fi	finishes	f
contains	di	during	d
starts	s	started by	si
equal	$=$	equal	$=$

In a system with a large number of intervals, there is always a chance they overlap. Two intervals must always satisfy the *interval trichotomy* that exactly one of the following three properties holds [100] :

1. i and i' overlap (i.e. $i \cap i' \neq \phi \equiv t_1 \leq t'_2$ and $t'_1 \leq t_2$)

2. i is to the left of i' (i.e., $t_2 < t'_1$),

3. i is to the right of i' (i.e., $t'_2 < t_1$)

Figure 3.4 shows the interval trichotomy for two closed intervals i and i'.

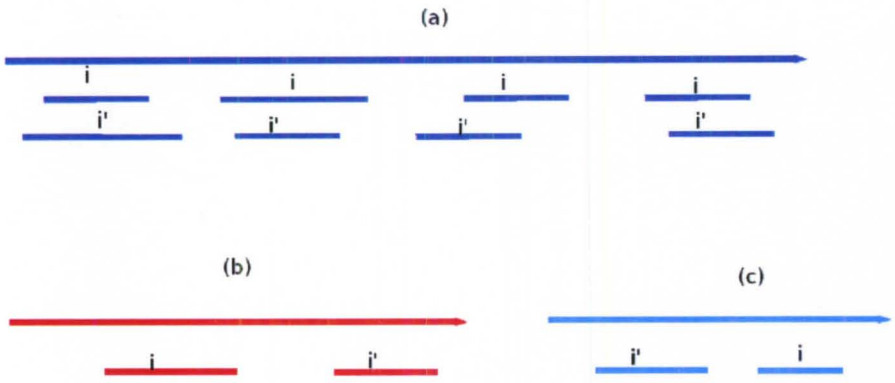

Figure 3.4: Interval trichotomy

3.3 Interval trees

There are a number of data structures and associated algorithms to deal with intervals [100–103]. Finding overlapping intervals is an output sensitive algorithm and depends on the number of inputs n (query intervals) as well as the the outputs m (intervals that map to an interval). Considering the fact that GEs are large in number, overlap with many other GEs, and are highly dynamic, an efficient data structure is needed for storage, information retrival, and update.

A *red–black* tree is a binary search tree having an extra attribute: the *color*, the value of which is either red or black. Other than the requirements imposed on binary search trees, a red–black tree follows the following properties:

- Every node is either red or black.
- If a node is red, then both its children are black.
- The root node is black.
- Every simple path from a node to a descendant leaf contains the same number of black nodes.
- Every leaf node (sometimes called sentinels) is black.

Each node of red–black tree contains the attributes color, key, left, right, and p. If a child or the parent of a node does not exist, the corresponding pointer attribute of the node contains the value

NIL. The constraint on the color of the nodes make the tree approximately balanced by ensuring that, no simple path from the root to a leaf is more than twice as long as any other. The height of a red–black tree is at most $2 * log_2(n+1)$, where n is the total number of nodes. Since operations such as inserting, deleting, update and finding values require worst–case time proportional to the height of the tree, this theoretical upper bound on the height allows red–black trees to be efficient in the worst-case, unlike ordinary binary search trees.

An *interval–tree* is an augmented red–black tree that maintains a dynamic set of elements, where each node i contains an interval storing the two endpoints $t_1[i]$, $t_2[i]$ and $max[i]$, which is the maximum value of all right endpoints in the subtree rooted at i.

$$max[i] = max(t_2[i], max[left[i]], max[right[i]]).$$

Here, $left[i]$ and $right[i]$ represents the left and right child of node i respectively. Nodes may store additional information. These nodes are ordered according to the low endpoint of the intervals and the inorder traversal of the tree always gives a sorted list. An example interval tree is shown in Fig. 3.5 which is constructed from the intervals shown at the bottom of the figure. Each node contains end–points of an interval and max (as described in text). The entry in the root node represents the interval with low–end point as 17, high–end point as 22 and max value as 29.

An interval tree allows dynamic insertion, deletion and search operations to be performed efficiently. Nodes are inserted and deleted in such a way that the properties mentioned above are always followed. The running time for all three operations is $O(log_2 n)$, and the updating of max can be done in $O(1)$ time. When there are multiple intervals (m) that overlap a query interval, the run time is $O(m + log_2 n)$. The preprocessing time to construct the tree is $O(nlog_2 n)$, with a space complexity of $O(n)$ [100].

The following operations can be performed on an interval tree:

(a) INTERVAL–INSERT(T, x): add an interval x into the interval–tree T. The running time for this operation is $O(\log_2 n)$.

(b) INTERVAL–DELETE(T, x): delete an interval x from the interval–tree T. The running time for this operation is $O(\log_2 n)$.

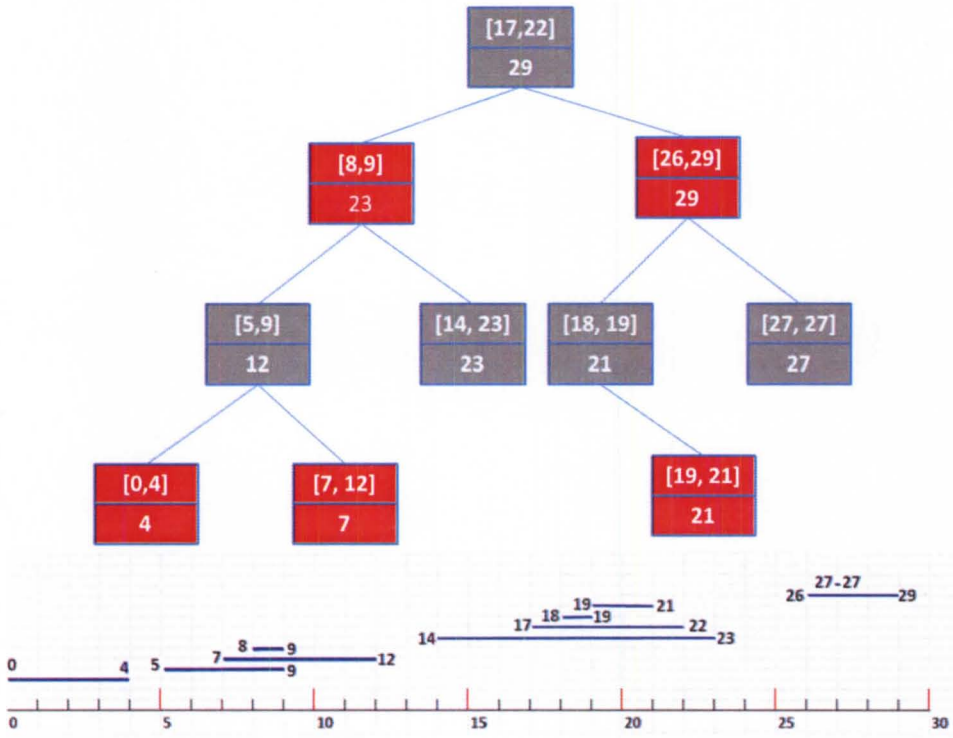

Figure 3.5: Example interval tree.

(c) INTERVAL–SEARCH (T, i): return a pointer to a node x in T such that int[x] overlaps interval i. INTERVAL–SEARCH (T, i) procedure is shown in Algorithm 1. If there is no interval that overlaps i in the tree, then the procedure returns *nil*. The running time for this operation is $O(\log_2 n)$. if there are multiple intervals that overlap i, then the running time will be $O(k + \log_2 n)$, assuming that there are k overlapping intervals. In case of genomic annotations, finding multiple overlaps is frequent because a particular region on a genome may be annotated at different granularity level, and by different authoritative organizations.

The preprocessing time to construct the tree is $O(n \log_2 n)$, with a space complexity of $O(n)$. The *interval–tree* is a special data structure to deal with the type of problem we are concerned about, since the number of genomic annotations are huge and require frequent insertions and deletions.

Algorithm 1 Algorithm for searching interval i in tree T [100].

```
1: procedure INTERVAL–SEARCH(T, i)
2:     x ← T.root;
3:     while x ≠ T.nil and i does not overlap x.int do
4:         if x.left ≠ T.nil and x.left.max ≥ i.low then
5:             x = x.left
6:         else
7:             x = x.right
8:         end if
9:     end while
10:    return x
11: end procedure
```

3.4 Using interval trees for finding overlapping GEs

Finding all genetic entities (GEs) in a given region of a particular genome is a common task in high-throughput molecular biology experiments. Considering the large number of available annotations and the fact that the structural annotations are dynamic (frequent insertions and deletions), an interval tree is implemented to store these annotations and perform various operations efficiently. A common operation is to find overlapping intervals. To accomplish this, sequence level information for different entities were downloaded from respective authoritative websites. Those considered include Affymetrix® and Agilent microarray probes, Entrez genes [88], EST sequences [104], and Ensembl transcripts [83]. Reference genomes were obtained from the UCSC Genome Browser website (rat version 3.4, mouse version mm9 and human version hg19). These annotations were then mapped onto their respective organism's reference genomes using either the BLAT [34] (GE sequence length $>$ 100 bases) or Bowtie [35, 36] ($\leq$ 100 bases) sequence alignment algorithms. For Bowtie alignments, the maximum number of mismatches allowed was two. Each alignment annotation includes t_Start (the start coordinate on the genome), t_End (end coordinate on the genome), size (of the mapped region), and chromosome number. Organism and annotation specific interval trees are maintained. *IRanges* [105] is used to incorporate the interval trees. The total number of GEs mapped from rat, mouse and human genomes are 34.1 million for this test set. These interval trees can be then queried for overlapping intervals (annotations). The design flow is shown in Fig. 3.6.

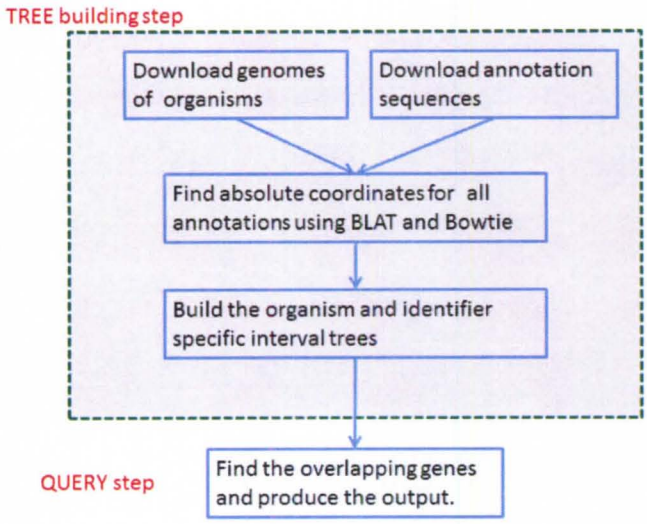

Figure 3.6: Steps to find overlapping annotations

3.5 Results

To demonstrate the efficiency of the interval tree implementation, random samples of 10,000 (10k), 50k, 100k, 500k and 1 million human EST accessions were mapped against themselves to find all overlapping ESTs. The number of nodes in the tree is increased exponentially adding one level in the tree at each increment. When the tree is small, the mapping time is less than a second. The number of overlapping identifiers and elapsed time were calculated five times and averaged. Fig. 3.7 shows the average overlap time plotted against the number of nodes in the interval tree. It took 27.3 seconds to map 10,000 nodes against an interval tree containing 8.27 million nodes, whereas 500,000 ESTs were mapped in 3.8 hours (not shown). The number of overlapping ESTs for 1 million ESTs was not considered due to memory constraints. The plot shows the run time to be linear as the number of nodes in the tree increases exponentially. One million randomly sampled EST identifiers in rat were also mapped to a total of 271.3 million overlapped ESTs in 276.5 seconds.

Fig. 3.8 shows the average number of overlapping EST identifiers as the number of nodes in the interval tree is increased exponentially. EST intervals with total number of inputs 10k, 50k, 100k, 500k and 1 million are given as input to find average number of overlaps. Mapping 1 million ESTs against an interval tree of size 4.19 million resulted in 787.9 million overlapping EST intervals.

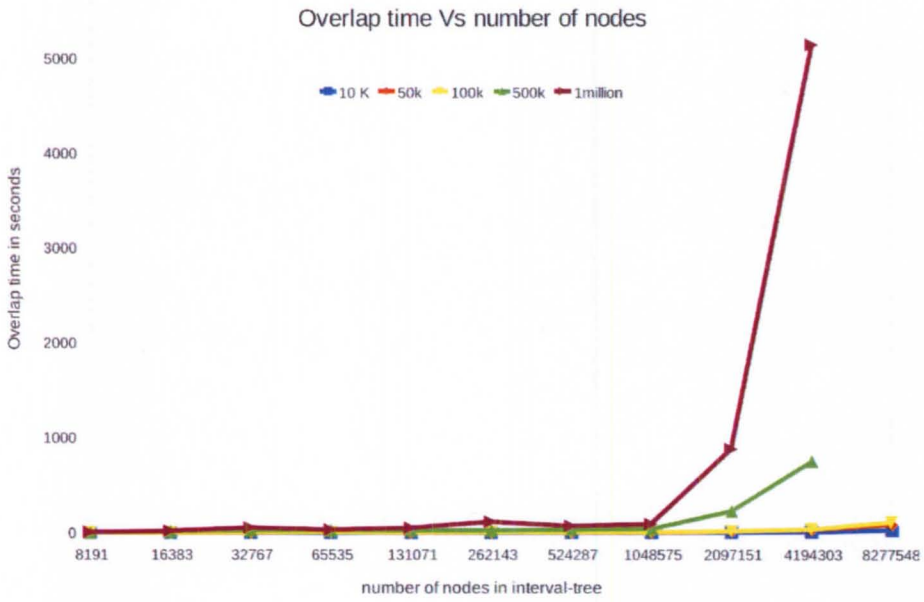

Figure 3.7: Average elapsed time for mapping ESTs.

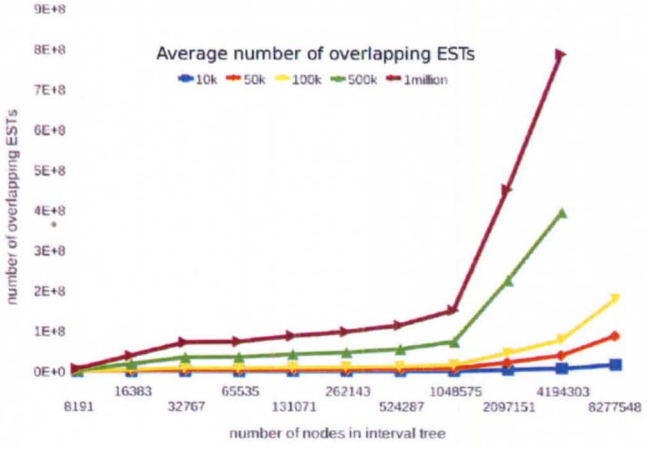

Figure 3.8: Average number of overlapping EST intervals.

An interval tree method was implemented and used to query these interval information. For comparison with relational databases, an equivalent MySQL database was also implemented to perform ID conversion based on coordinate information, and the run time for both of these methods were compared. Fig. 3.9 shows the run–time comparison of the interval tree and MySQL implementation when randomly sampled EST IDs from rat are converted to Entrez gene IDs. The number of EST IDs was increased exponentially for each test point and the corresponding time in seconds was measured. The run time using the interval tree takes negligible time to map hundreds of thousands of overlapping genes as compared to the relational database method. The execution time increases rapidly in the case of MySQL as searching intervals is a linear process. Conversion of one million EST identifiers into Entrez using the interval tree method took just 11.28 seconds.

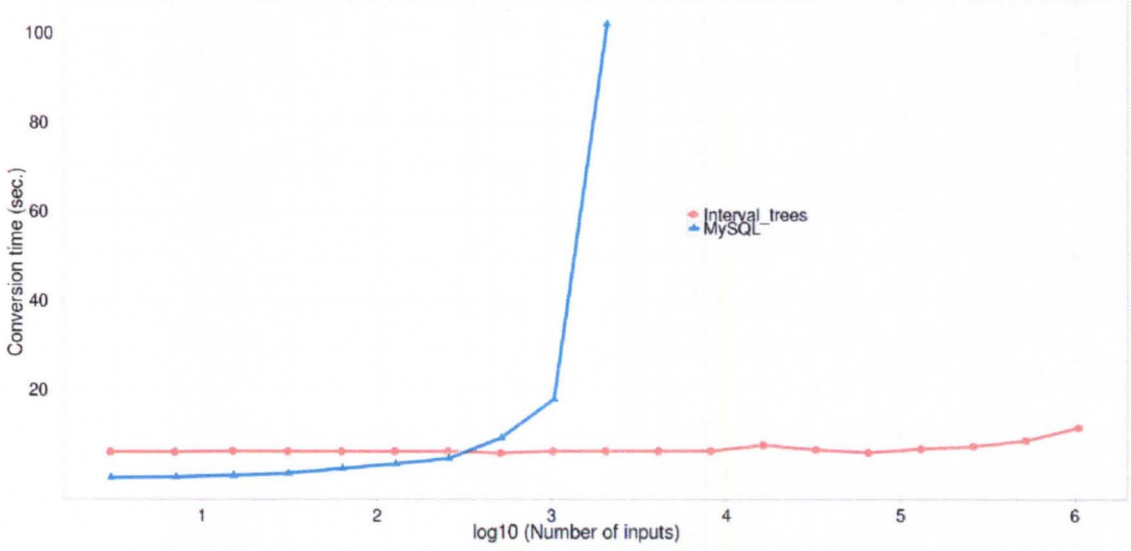

Figure 3.9: Run time comparison for converting EST IDs into Entrez Gene IDs.

Finally, Affymetrix® HG-U133 Plus 2.0 probes were mapped onto the human genome and the corresponding intervals determined. These intervals were then queried against the Agilent CGH 105a, GenBank ESTs, and Ensembl transcripts intervals. The number of overlapping intervals and the elapsed time are shown Table 3.2. Here n represents the number of intervals being queried. The number of nodes in the interval tree for Agilent CGH 105a, EST and Ensembl transcript is 206712, 8277548, and 151222 respectively. Querying 1.02 million intervals from the Affymetrix® probe on

EST took 42.94 seconds. All of the overlapped Ensembl transcripts and Agilent probes were found in 3.9 seconds.

Table 3.2: Number of overlapping intervals and overlapping time (sec.)

	CGH105a		EST		Ensembl	
Size(n)	**Overlap**	**time**	**Overlap**	**time**	**Overlap**	**time**
2	0	4.63	2258	26.84	318	3.77
4	26	3.74	3117	27.97	136	3.48
8	0	3.55	4186	25.28	265	3.46
16	4164	5.51	1862002	30.93	254878	3.92
32	3146	3.75	1681155	29.28	195395	4.52
64	5288	3.68	2546796	26.24	325845	4.66
128	5916	4.68	3032338	29.4	372675	4.02
256	6234	4.61	3769694	31.63	395812	5.1
512	7790	4.67	4884356	31.85	485259	4.05
1024	8086	3.72	5695468	33.14	500769	4.92
2048	9018	4.59	7100439	31.43	568468	4.95
4096	11164	4.65	10268391	31.44	672567	6.05
8192	14288	5.47	13732435	30.84	801939	5.09
16384	18690	7.46	21452652	39.53	1016474	5.19
32768	25670	6.69	29380179	44.81	1344805	4.47
65536	35078	4.91	41891464	54.02	1794852	4.65
131072	46740	4.85	54268926	59.75	2314813	4.85
262144	54514	4.01	61601695	50.14	2673404	5
524288	56292	4.03	63060840	51.52	2753457	5.06
1026588	56302	3.89	63306821	42.94	2754880	3.91

3.6 Conclusion

An interval tree was implemented as an augmented red-black tree for storing and querying the genomic structural coordinates of GEs. The results demonstrate that an interval tree is a better alternative for maintaining data that represents intervals by providing queries that grow in logarithmic time with respect to the number of annotations present as opposed to the linear growth of relational approaches. Interval trees serve as a dynamic data structure that can handle insertion, deletion and search operations efficiently. Representing genetic intervals is one of numerous applications where interval trees have an edge over contemporary methods in terms of efficiency. These techniques are readily applicable to others applications such as database transactions, weblogs, and others where the number of intervals run into tens of millions. Insertion and deletion in an interval tree can be performed efficiently in $log_2 n$ time whereas finding overlaps can be performed in $O(m + log_2 n)$ time

with a space complexity of $O(n)$. The interval tree method is limited by the fact that it needs to be in memory all the time to perform interval queries. This may not be as big of a concern as the memory size in modern computer systems is typically large enough to hold these annotations. However, if data can not fit into main memory, a method such as that proposed by *Arge et al.* [102] [103] can be used that maintains the interval tree in secondary memory efficiently. The power of interval trees for querying annotations in genetic entities will prove useful in the context of the information explosion from high throughput molecular biology technologies such as next generation sequencing, proteomics and metabolomics.

CHAPTER 4

ABSIDCONVERT: AN APPROACH TO CONVERT GENETIC IDENTIFIERS AT DIFFERENT GRANULARITIES

4.1 Introduction

The Nucleic Acid Research (NAR) 2012 database issue [2] features 1380 databases covering various aspects of molecular biology including sequences, gene expression, structures, pathways and diseases. Most of these databases are independent of each other and have been created as a result of the respective developers' domain of interest and resource limitations. Due to a lack of standard naming conventions, most of these databases prefer to assign their own custom generated identifiers (IDs) to the biological entities. Major public databases such as GenBank [85] and RefSeq [86] use accession numbers, Gene Ontology (GO) [82] uses a naming convention from organism specific databases, the HUGO (Human Genome Organization) Gene Nomenclature Committee (HGNC) [84] uses the gene symbol and a custom generated ID, Entrez [88] uses numeric integers, sequencing projects use systematic names and biologists sometimes use additional aliases. As an example, the breast cancer early onset gene has the official gene symbol of BRCA2 provided by HGNC and an associated ID 1101, Ensembl [83] gene ID ENSG00000139618, OMIM (Online Mendelian Inheritance in Man) [106] ID 600185, HPR (Human Protein Reference) database [107, 108] ID 02554, RefSeq ID NM_000059, GenBank Accession U43746, Entrez Gene ID 675, VEGA (the Vertebrate Genome Annotation database) [109] gene ID OTTHUMG00000017411, UCSC [92, 98] gene ID uc001uub.1, UniProt [87] ID P51587, and gene aliases FAD, FAD1, BRCC2, FANCD1, FACD, FANCD.

Fortunately, there is a wealth of information available to the research community in a wide variety of databases. However, it is often difficult to extract or integrate information about a

particular biological entity from multiple resources. For instance, a researcher may be interested in extracting functional information spread across different databases for a biological entity such as a gene or a protein; comparing two independent pathways which use different types of identifiers; or comparing results across species, platforms or labs. The lack of a common identifier across these heterogeneous and sometimes redundant biological databases makes the functional analysis of biological data tedious, time consuming, and error prone.

One solution to handle heterogeneous databases is to use a global identifier for annotations such as the one described by MIRIAM (Minimum Information Requested In the Annotation of biochemical Model) [110]. MIRIAM requires a global identifier to contain both the data source as well as an internal identifier. For example, *urn:miriam:hgnc:brca2* is composed of *urn:miriam* that defines the notation to be a URN (Uniform Resource Name) using the MIRIAM scheme with data type *hgnc* and identifier *brca2*. This method appears promising and has the potential to solve some of the previously mentioned problems, but very few databases follow this standard. Another solution is to manually search for these genes one by one in publicly available databases such as Entrez, KEGG [111, 112], or GEO [93, 94] and infer their functionality. This method is fruitful when the number of genes is small, but is impractical for high throughput experiments, where the number of gene fragments can be on the order of tens of thousands or more. A third solution is to use an ID converter tool that uses a database to store all possible annotations where a list of IDs may be input as a query which is then converted into the corresponding target IDs in a precise and efficient way.

One difficulty in the development and maintenance of such conversion tools is the varying granularity of the identifiers. More specifically, the data generated by biological experiments may be at the locus, transcript, sequence or probe level, with varying coverage of a region of interest (Fig. 4.1). This granularity ranges from very fine, at the level of DNA microarrays (tens of bases in length, containing probe level information relevant to only a short region of the corresponding mRNA molecule) through coarser granularity with sequence reads (few hundreds), transcripts (thousands), loci, and chromosomes. It is also possible that annotations at the same level may have different granularities. For example, among DNA microarray probes, Affymetrix® probes are usually short (25 bases)

whereas Agilent probes are longer (60 bases) and cDNA probes are generally $\geq$ 500 nucleotides in length. The relationships between entities at the same or different granularities may be either 1-1, 1-n, n-1 or n-m: for example, an Affymetrix® probe may span more than one EST; more than one such probe may be contained inside an EST; a cDNA probe may contain zero, one or more Affymetrix® and Agilent probes.

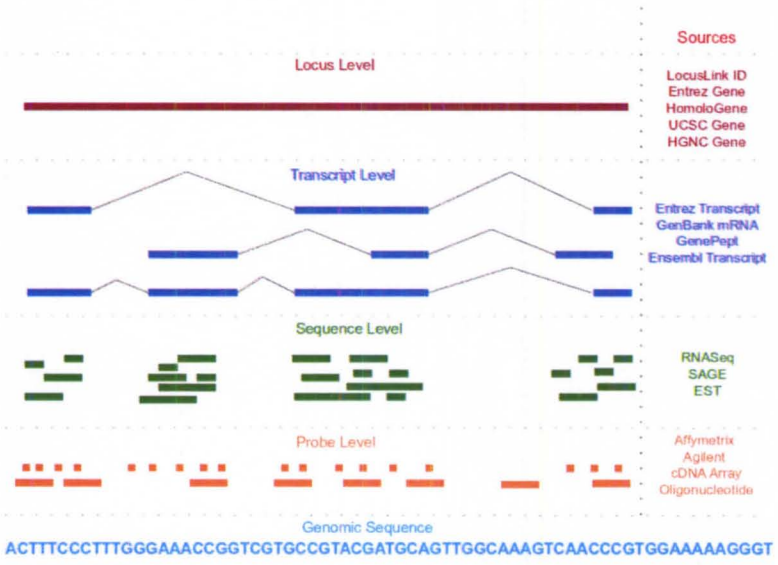

Figure 4.1: Granularity of annotations

Another difficulty in the development of such tools is the dynamic nature of annotations. Of late, rapid advances in sequencing and their declining costs have enabled researchers to perform novel sequencing as well as resequencing projects. These result in an increased depth of coverage of a genomic sequence, with gaps being filled and repeats more accurately mapped. Sometimes, the sequence underlying a genetic entity may change, and on a less frequent basis the whole genomic sequence needs to be updated (as of April 15^{th}, 2012, the currently available genome versions for human, mouse and rat are 19 (GRCh37), 10 (GRCm38) and 4 (RGSC v3.4) respectively). These changes may modify the structural and functional annotations of a genetic entity (GenBank, RefSeq and Ensembl are updated everyday). Frequent updates in annotations also create problems in the manufacturing of DNA microarrays. Microarray chips are designed and their probes are annotated using the current build of a specific genome. Regardless of the care taken in this design, the

system will include flaws due to the combination of the delay inherent in the process of microarray design-manufacture-deployment (compounded by the latency to use) and the dynamic nature of annotations. Attempts to address these problems have been the focus of a number of previous studies. Gautlier et al. [113] found redundancies in the annotations of Affymetrix® probes at a sequence level that map to multiple RefSeq genes. Such ambiguities may result in inaccurate interpretations. AffyProbeMiner [114] uses RefSeq and GenBank's validated complete coding sequences to regroup the probes on an Affymetrix® chip into consistent probe sets. In their study, regrouping of the probes affected almost 65% of the probes on the HG-U133A chip. Harbig et al. [115] reidentified the Affymetrix® U133 plus 2.0 GeneChip® array probes in an attempt to increase the reproducibility of microarray experiments. They used BLAST [33] to remap the probes against the genome and redefined approximately 37% of the probes. These studies suggest that redefinition or reorganization of probesets will improve the analytical accuracy of the microarray data, a process that would be greatly facilitated by a means for high-throughput query and mapping/comparison of given sequences (such as microarray probes) to other genomic annotations stored across a wide variety of databases.

4.2 Currently available ID conversion tools

The problem of ID conversion persists even though a number of tools exist to address this problem. Some of these are generic and perform ID conversion for probes, genes, proteins, and additional annotations while others are more specific to DNA microarray probes. Organism support varies with many of the tools catering to either a single organism or a small set of comparable species. In addition, cross–species comparison is variable, with most methodologies providing only intra–species conversion. Almost every approach uses some sort of relational database with the unique identifier being Ensembl IDs, RefSeq IDs, or custom generated IDs. A brief description of some popular tools follows.

DAVID (Database for Annotation, Visualization and Integrated Discovery) [116–118] is a web based structural and functional annotation tool to extract biological meaning from a gene list. It uniquely generates custom IDs for querying a set of relations and is dependent on annotations from

other databases. A component of *DAVID*, *DICT* [119] (DAVID gene ID Conversion Tool), facilitates ID conversion. *EASE* [120], developed by the DAVID Bioinformatics team, is a customizable, standalone, Windows® desktop software application, having similar analytic capabilities as that of *DAVID*. *Babelomics* [121, 122] is an integrated web based tool for structural and functional annotation with an *ID converter* being one of its components. This component uses a universal index linked to Ensembl to create a database of 11 species. *g:Convert* [123], a component of *g:Profiler*, allows arbitrary conversion of genes, proteins and probes into one another. Every alias in *g:Profiler* is mapped through a three-level index of gene, transcript and protein Ensembl IDs. For each index level, all corresponding IDs are stored in the database. The *Hyperlink Management System and ID Converter System* [124] automatically updates and maintains hyperlink information among major public biological and chemical databases. It downloads data everyday from authoritative databases and produces a large correspondence table which is used to show the most up-to-date URL for genes of interest. Users can use CGI programs to create hyperlinks to this data. *Synergizer* [125] assigns a unique internally generated identifier, "peg", to all external IDs that refer to the same biological entity. It mostly uses the NCBI "gene2accession" file to maintain a database of synonym relationships and produce a simple web interface. *MADGene* [126] uses correspondence tables and allows conversions in an efficient way. The *Clone/Gene ID Converter* [127], *MatchMiner* [128], the *Gene name converter* in *GeneMerge* [129], *RESOURCERER* [130] and *GeneLynx* [131] are additional ID conversion tools.

Some of the ID conversion tools are more specific, such as those that work only at the probe level. *GATExplorer* [132] is a web based tool for analysis and visualization of Affymetrix® probes at the genomic and transcriptomic level. It performs de–novo mapping of all the probes of Affymetrix®'s expression and exon arrays against the transcriptome of the corresponding organism using BLAST and records the coordinates on the genome. Unmapped probes are mapped to an ncRNA database downloaded from RNAdb. Only the perfect match alignment is selected while mapping these probes. The location of a gene or probe on the genome can be visualized along with all the transcripts present in that region. *NetAffx*™ [97], provided by Affymetrix®, performs ID conversion of Affymetrix®

probes for different organisms and has a feature to perform structural and functional annotation. *PLANdbAffy* [133] is a Probe-Level ANnotation database for Affymetrix® microarrays (HG-U133A, HG-U133B, HG-U133 plus 2.0, Human Exon 1.0, Human Gene 1.0) that uses BLAT [34] to map individual probes onto the human genome. These probes are then annotated using information extracted from RefSeq. *ProbeMatchDB* [134] uses a number of public databases to perform cross-species and cross-platform probe mapping. The database conversions are enabled by UniGene and HomoloGene identifiers. UniProts [78, 135] *ID mapping tool* works on the gene and protein level and converts gene IDs into UniProt IDs and vice versa.

Some software tools have unique methods for mapping between different IDs. *Onto-Translate* [60, 136] converts one type of IDs into another by calculating the optimal path between IDs, taking into account the "trustworthiness" of data contained in various databases. The *AliasServer* [137] uses a custom generated unique 64-bit reference identifier which is computed from the amino acid sequence using the CRC (Cyclic Redundancy Check) algorithm where each ID is a unique combination of species identifier, type of database and the ID itself.

Some databases/tools aid in ID conversion but do not function as a full fledged ID conversion tool. *BioMart* [138, 139], earlier known as *EnsMart* [140], provides a web and API interface to download data such as GO terms, genes, transcripts and expression arrays from different databases using filters. *BridgeDb* [141] provides an interface to connect bioinformatics tools such as Cytoscape, PathVisio, or WikiPathways with other mapping services such as Ensembl, PICR (Protein Identifier Cross-Reference services) [142], and any local database or text files. It is intended to be used by bioinformatics developers and works on the novel idea of mapping custom identifiers to established identifiers such as Ensembl ID and then relies on Ensembl to provide the rest of the conversion. Side by side feature comparisons of these tools are provided in Table 4.1. Data sources for select tools are listed in Table 4.2.

Table 4.1: Feature comparison of different conversion tools.

Name	Caters to	intervals to IDs	seqs to IDs	ID lookup	Annot. View	linkout	Query mode	Input	Output	Annot.	Basis of conversion	output format	Organisms	availability	Last update
DAVID	probes, genes, prots.		✓	✓	✓	✓	batch	select one	select one	S, F	custom generated	html, txt	NA	web,API, EASE, ⇓	Sep, 2009
Babelomics	probes, genes, prots.				✓	✓	batch	select one	select multiple	S, F	custom generated	html, txt	11 org.	web	Sep, 2009
g:Convert	genes , prots. and probes						batch	select one	select one	S, F	Ensembl	html, txt, xls	H, M, R, O	web	Jun, 2011
HMS and IC	genes, prots. and bio. molecules				✓	✓	batch	select one	select one	S, F	corr. files	html, txt	H, M, O	web, ⇓	current
Synergizer	probes, genes and Prots.						batch	select one	select one	S	Peg/custom generated	html, xls	H, M, R, O	web, API	May, 2011
Clone/Gene ID Converter	genes and prots.					✓	batch	select one	select multiple	S, F	Ensembl	html, txt, xls	H, M, R	web	Apr, 2008
MADGene	probes, genes, trans.					✓	batch	NA	select multiple	S, F	MADGene link	html, xls	H, M, R, O (17 org.)	web, open source	Aug, 2009
GATExplorer	Affy expression & exon arrays		✓		✓	✓	single	probes	genes, trans.	S	Ensembl	html	H, M, R	web, ⇓	Mar, 2010
NetAffx™	genes, prots., probes, other				✓	✓	batch	select one	select one	S, F	UniGene, LocusLink	html, txt	H, M, R, O	web	CND
PLANdbAffy	Affy expression arrays				✓	✓	single	Affy, Hugo,Ens	Affy, Hugo,Ens.	S	RefSeq	html	H	web, ⇓	May, 2009
probeMatchDB	probes, cDNA, EST, gene, prots.			✓		✓	batch	select one	select one	S	UniGene, Homologene	html	H, M, R	web	2006
Uniprot	genes and prots.				✓	✓	batch	genes or prots.	prots. or gene	S, F	UniProt ID	html	NA	web, API, ⇓	Jul, 2011
Onto-Translate	Affy, uniGene clusters, Acc num					✓	batch	select one	select one	S, F	RefSeq, Entrez	html, email	H, M, R, O (58 org.)	web	May, 2009
AliasServer	Affy, genes, prots.					✓	batch	select one	select multiple	S	custom generated	html, txt	Not Available	Not Available	CND
MatchMiner	Affy, genes			✓			batch	select one	choose from	S	custom generated	Email (txt,xls)	H, M	web	Sep. 2006
GeneMerge	genes and prots.						batch	select one	NA	S, F	corr. files	html	5 org.	web	Apr, 2007
BioMart	genes, prots., probes, other				✓	✓	NA	select one	select multiple	S, F	NA	html, txt, xls	H, M, R, O	web, API, ⇓	depends on DB
BridgeDb	probes, genes, prots., metabolites			NA	NA	✓	NA	NA	NA	S, F	Ensembl, other	NA	36 org.	open source	May, 2011
AbsIDconvert	genes, trans., seqs., probes	✓	✓	✓	✓	✓	batch	select one	select multiple	S	Genomic Sequence	html, txt	H, M, R, O (53 org.)	web, ⇓VM	Dec, 2011

Abbreviations: Annot. View: Custom Annotation view, Annot.: Annotation (S: Structural annotation, F: Functional annotation), org.: Organisms (H: Human, M: Mouse, R: Rat, O: others), prots: proteins, Affy: Affymetrix®, trans: transcripts, seqs: sequences, Ens.: Ensembl, corr: correspondence, acc: accession, bio: biological, NA: Not Applicable, ?: Could not determine, ⇓:download Knowledgebase, VM: Virtual Machine.

Table 4.2: ID converter tools, data sources and availability.

Name	Data Sources	Webpage
DAVID	GenBank, RefSeq, KEGG, OMIM, UniGene	http://david.abcc.ncifcrf.gov/
Babelomics	Go, KEGG, Ensembl and others	http://babelomics.bioinfo.cipf.es/
g:Convert	GO, KEGG, Ensembl, TRANSFAC, Reactome	http://biit.cs.ut.ee/gprofiler/
HMS and IC	Ensembl, GO, KEGG and others	http://biodb.jp/
Synergizer	Ensembl, NCBI, RGD, SGD, KEGG, WormBase and EcoCyc	http://llama.mshri.on.ca/synergizer/translate/
Clone/Gene ID Converter	Ensembl, NCBI, Pubmed, UCSC, KEGG, Reactome	http://idconverter.bioinfo.cnio.es/
MADGene	GEO, UniGene, Entrez and others	http://www.madtools.org/
GATExplorer	Ensembl, Affymetrix®	http://bioinfow.dep.usal.es/xgate/principal.php
NetAffx™	NCBI, GO, KEGG and others	www.affymetrix.com/analysis/netaffx/
PLANdbAffy	Affymetrix®, UCSC, NCBI	http://affymetrix2.bioinf.fbb.msu.ru/
probeMatchDB	UniGene, HomoloGene	http://brainarray.mbni.med.umich.edu/Brainarray/
Uniprot	GenBank, RefSeq, GO and others	http://www.uniprot.org/
Onto-Translate	Ensembl, GO, KEGG and others	http://vortex.cs.wayne.edu/
AliasServer	Ensembl, EMBL, NCBI, SGD and others	http://cbi.labri.fr/outils/alias/
MatchMiner	Affymetrix®, UCSC, UniGene, Entrez, OMIM	http://discover.nci.nih.gov/matchminer/index.jsp
GeneMerge	GO, KEGG	http://genemerge.cbcb.umd.edu/
BioMart	NCBI, GO, KEGG and others	http://www.biomart.org/
BridgeDb	Ensembl and others	http://www.bridgedb.org/
AbsIDconvert	UCSC, NCBI, Ensembl, Agilent, Affymetrix® and others	http://bioinformatics.louisville.edu/abid/

4.3 Drawbacks associated with existing approaches

Most of the ID conversion tools mentioned above use a two step conversion method. To convert an ID A to ID B, the first step is to use a correspondence annotation relation or table to find a common intermediary ID C (Fig. 4.2). This common ID C is then converted into target ID B using another correspondence table. Some tools use Ensembl or RefSeq as an intermediary while others generate unique custom identifiers. For example, the Clone/Gene ID Converter and GATExplorer use Ensembl ID, PLANdbAffy uses RefSeq whereas DAVID and Synergizer use a custom generated DAVID ID and peg respectively. These tools convert smaller fragments (probes, sequences, reads) into coarser genetic entities (Ensembl, RefSeq, EntrezID) using inferred annotation level information irrespective of the fact that these small fragments may not be representative of the annotation as a whole. These methodologies also tend to lose structural and other information available at the probe or sequence level.

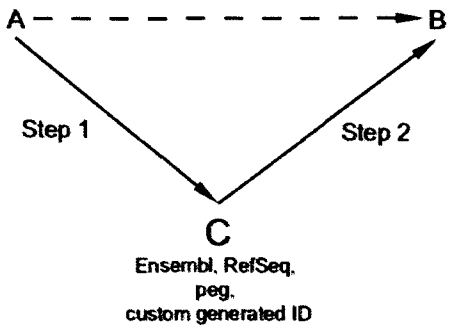

Figure 4.2: ID Conversion - A two step process.

As stated previously, annotations are dynamic and databases such as Ensembl and RefSeq are updated daily making it difficult to keep the databases of ID conversion tools current. This is more problematic when the intermediate IDs are custom generated as these require more effort to update. Most of the tools are based on a relational database and the dynamic nature of annotations may introduce database anomalies because of the frequent insertion, deletion and updating of the annotations. If a gene is discovered, deleted or updated in any of these databases, or the annotations corresponding to an entity are added, deleted or updated, then all the databases or correspondence

tables also need to be updated. In the case of microarray experiments, if a probe corresponds to a recently deleted entity then that probe annotation needs to be edited as well. Updating any of these authoritative databases may induce a chain-reaction for any other systems using that information and any experimental result deduced from the updated probe may become invalid. Those tools that generate their own unique identifier such as DAVID, Synergizer or Babelomics, although efficient, face a similar situation and need to be updated frequently. As updating an annotation database is labor and resource intensive, some of the tools cannot afford to update their knowledgebase regularly.

4.4 Absolute (sequence based) method for ID conversion

A feature of biological entities that is currently ignored in ID conversion is the sequence mapping information. For species where a reference genome is available, all nucleic acid and protein-based annotations, no matter the granularity, can be aligned to that reference genome sequence and therefore annotated by genomic intervals. Once the absolute genomic coordinates on a reference genome for all entities have been determined, these can be queried to find all overlapping entities, thus performing ID conversion. This conversion uses the same two step method as adopted by most of the ID conversion tools, considering the genomic coordinates as the basis of conversion, rather than the annotation level information used by other tools. Compared to other types of intermediate IDs, the intervals on a reference genome sequence are relatively static, and remapping of entities to modified genomic sequences is relatively trivial, making it possible to easily update the system. Using interval trees, conversion by finding overlapping intervals is fast and efficient [143].

Fig. 4.3 shows the steps to perform sequence-based or absolute ID conversion. In the figure, all transcripts corresponding to probe A are being found. The first step (step 1) in this conversion is to find the genomic coordinates corresponding to probe A and the second step (step 2) is to find all transcripts that span those coordinates. In this example transcript 2 and transcript 3 are the converted IDs corresponding to the probe A. Transcript 1 is not represented by probe A as the underlying genomic sequence is not part of transcript 1. Subsequent sections describe the design and implementation of a genomic interval based ID conversion tool, AbsIDconvert.

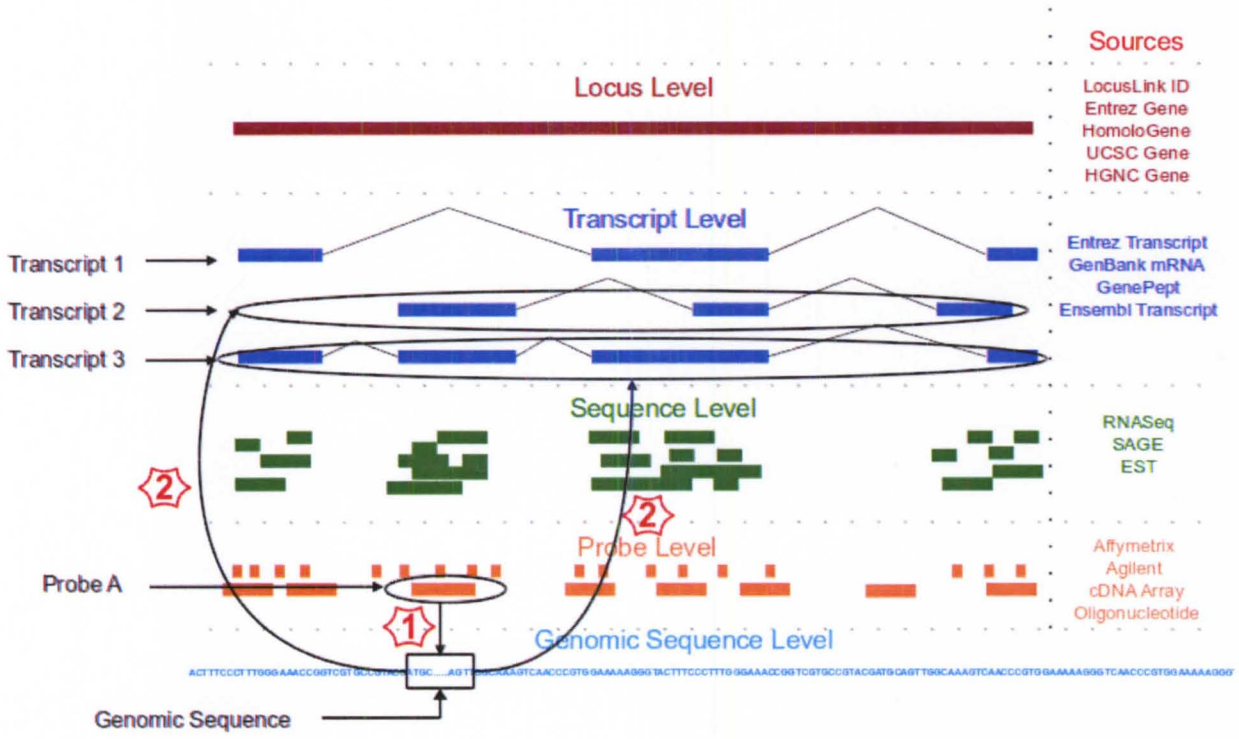

Figure 4.3: Absolute ID conversion process

4.5 System design and implementation

The design of AbsIDconvert was accomplished using a preprocessing and a query step. In the preprocessing step, reference genomes were downloaded from the UCSC Genome Browser (`http://hgdownload.cse.ucsc.edu/downloads.html`) and the NCBI website. The sequence information for a variety of identifiers at different granularities such as probes, sequences (ESTs), transcripts and genes were downloaded from their respective authoritative websites or UCSC. The identifier types include Affymetrix® probes, Agilent probes, EST sequences, Ensembl transcripts and Entrez genes. Each identifier sequence was mapped to the respective genome using either BLAT [34] or Bowtie [35]. BLAT was used to map longer (>100 BP) sequences, while Bowtie was used for relatively short ($\leq$ 100 BP) sequences such as Affymetrix® and Agilent probes. Each identifier was then annotated with structural information such as *start* (identifier's start coordinate on genome), *end* (the end coordinate on the genome), *size* (sequence size) and *chrom* (corresponding chromosome). This information was collected for each identifier as a genomic interval. Genetic entities with multiple exons such

as transcripts were treated differently as there are two ways in which these can be structurally annotated. One method is to use the extreme ends (i.e. start and end codons of the transcript) as their intervals including both the exons as well as intronic regions, or alternatively exclude the intronic regions and assume the transcript's genomic intervals are an assembly of genomic intervals of the participating exons (AbsIDconvert incorporates both). Finally organism and identifier type specific interval trees were constructed and stored. A list of all identifiers and their type was also stored in a relational database to facilitate batch look-up for the types of identifiers. Fig. 4.4 shows the design steps of AbsIDconvert.

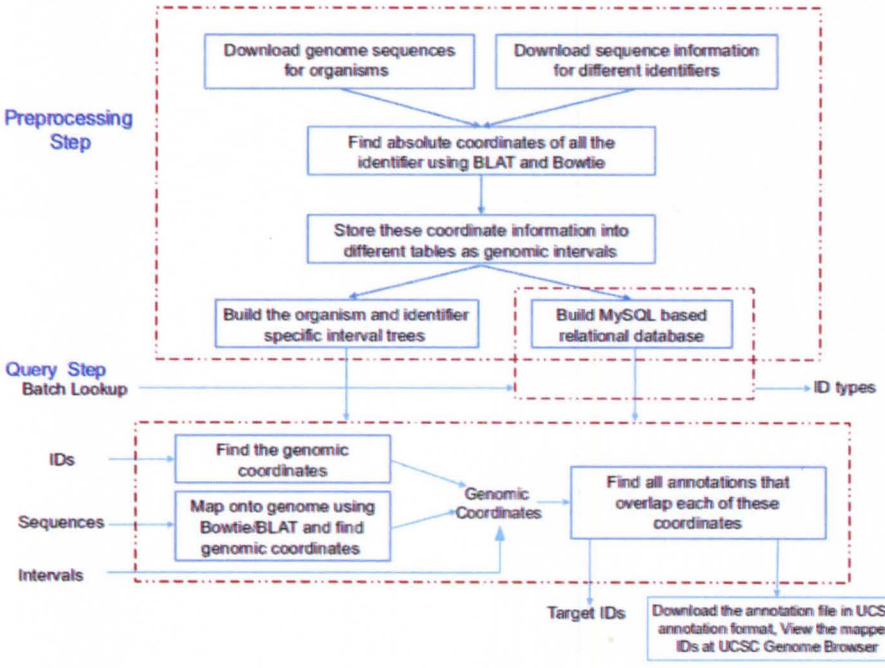

Figure 4.4: Steps involved in the construction of AbsIDconvert.

Once structural annotation for each of the identifiers is available, AbsIDconvert can query this information. This query step uses the structural annotation information of each identifier and the organism specific database generated from the previous step. AbsIDconvert assumes two biological entities (nucleic acid, protein entity) are the same if their genomic sequences are also the same, overlap or one is contained within the other. As the number of annotations are large and frequent insertions and deletions are routine, an efficient data structure for storage and computational operations is needed. Considering that the structural annotation is in the form of genomic intervals, a

modified Red-Black tree, known as an interval tree, is used to store the information for all IDs. An interval tree maintains a dynamic set of elements, with each element x containing an interval *int[x]*. This *int[x]* stores the start and end of the interval apart from other auxiliary information. This data structure is dynamic in nature and can perform insertions and deletions efficiently in time $O(log_2 n)$, where n is the number of elements. Interval trees have been shown to be efficient for working with a large number of genomic intervals as covered in Chapter 3

There are four possible ways in which AbsIDconvert may be queried:

- Lookup identifiers: Given a mixed list of identifiers, AbsIDconvert can determine the types of identifiers in the list. This step uses the relational database created in the preprocessing step and can efficiently categorize the IDs in the list.

- Batch conversion of IDs: Given a list of identifiers, AbsIDconvert uses the interval tree to find their genomic coordinates. Once the coordinate information is available, all overlapping identifiers can be found by querying the interval tree. This uses the IRanges [144] and GenomicRanges [105] packages internally to maintain the genomic intervals which are based on Allen's Interval Algebra [99]. Users can specify various range parameters using the interface. The overlap type ('type') parameter may take any one of 'any', 'start', 'end', 'equal' or 'within' as its value. By default 'any' overlap is accepted. If 'type' value is 'start' or 'end' then the query intervals are required to have matching 'start' and 'end' respectively with subject intervals in the database. If 'type' is 'equal' then only those subjects are retrieved which have the exact same coordinates. For 'within', the query must be contained wholly within the subject intervals. Another parameter is for specifying the maximum gap ('maxgap') between subject and query intervals to consider them as overlapping. The default value is zero which assumes there should not be any gap between the subject and query intervals. This parameter is useful for finding genes in the flanking regions of the specified intervals. The third parameter is the minimum overlap ('minoverlap') size that specifies the minimum number of overlapping base pairs needed to consider the query and subject an overlap. The default overlap value is one. The last parameter is the 'select' parameter that specifies which type of overlaps will be reported.

By default, all overlapping intervals will be reported. Selecting 'first', 'last' and 'arbitrary' will report first, last and arbitrary overlapping intervals from the result. A simple example using intervals is shown in Fig. 4.5. In this case, the reference genome is 10 BP long. The subject database contain four intervals s1, s2, s3 and s4 that represent the interval database. Query intervals also consist of four intervals q1, q2, q3 qnd q4. Considering default values for range parameters, q1 overlaps with s1, q2 and q3 overlap with all the intervals in the subject, whereas q4 overlaps with s2, s3 and s4. If the values of the parameters are type='within', maxgap = 0, minoverlap=1, select= 'all' then q1 overlaps with s1, q2 with s2 and q4 with s2 and s3. If the values of the parameters are type='end', maxgap = 1, minoverlap = 1, select= 'all' then q2 overlaps with s2, q3 with s3 and q4, and q4 with s2.

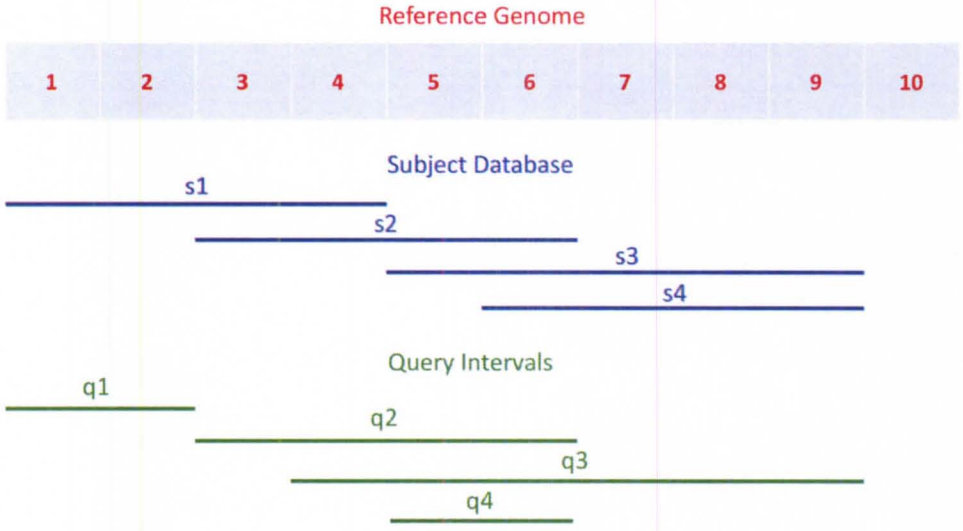

Figure 4.5: Example of interval overlaps.

- Intervals as input to AbsIDconvert: A unique feature of the ID conversion is to find target identifiers corresponding to a given interval. For example, next-generation sequencers generally map the DNA sequences or reads to a reference genome and output the intervals for each aligned reads. Finding desired target identifiers corresponding to these intervals is routinely required. AbsIDconvert efficiently converts these coordinates into target identifiers in a high throughput manner. For instance, a user of AbsIDconvert is able to take a set of intervals upstream of

a set of transcription start sites to determine if any features are annotated proximal to the regions of interest.

- Sequences as input to AbsIDconvert: Sometimes a user may be interested in finding all identifiers that correspond to a particular sequence or a list of sequences. For instance, a user may be interested in finding all gene names and Entrez IDs corresponding to a set of sequences. In this case, AbsIDconvert maps these sequences to the corresponding genome (or any other genome for cross–species comparisons) and determines the genomic intervals they belong to and then retrieves all the desired target identifiers that overlap these intervals. Due to the computational complexity involved in mapping long sequences using a generic mapping algorithm such as BLAT or BLAST, the web version of AbsIDconvert supports only short sequence mapping using Bowtie. Longer sequences can be mapped using BLAT in the virtual machine version of AbsIDconvert. Sequence output from next-generation sequencing technologies can be catered efficiently using AbsIDconvert. Alternatively, the coordinate information may be obtained by submitting the sequences to Galaxy [145–147] or the UCSC genome browser and subsequently inputting the intervals using AbsIDconvert. Mapping parameters can be specified by the user through the interface. Parameters include the maximum number of mismatches which can range from zero (default) to three. The second mapping parameter specifies which type of alignments are to be reported. The default value is 'all Best' in which all best alignments will be reported by Bowtie. However, 'all', 'k' or 'k Best' can be selected for Bowtie output. AbsIDconvert also has another parameter 'Do not report (..more)' that takes a positive integer value which specifies that Bowtie will suppress all alignments for a particular read if the total number of reportable alignments for that read is more than the specified value. The default value of -1 specifies that all alignments will be accepted. For instance, if this value is set to 100, then Bowtie will suppress all those alignments for reads that map to 100 or more locations on the genome. This is an effective option to mask repeat sequences or small sequences from appearing into the output because their probability to map at multiple locations on the genome is higher.

AbsIDconvert supports 53 major species for performing ID conversion on a list of identifiers and a list of intervals. It also has sequence level mapping support for 12 major species including *Homo sapiens, Mus musculus, Rattus norvegicus, Bos taurus, Gallus gallus, Sus scrofa, Xenopus tropicalis, Anopheles gambiae, Drosophila melanogaster, Caenorhabditis elegans, Saccharomyces cerevisiae*, and *Danio rerio*. AbsIDconvert converts the input (intervals, IDs and sequences) into target identifiers with links to authoritative databases. All intermediate interval files are available to download for later use. It also generates custom annotation files that can be used to view the IDs simultaneously (chromosome–wise) as a custom track in the UCSC Genome Browser. The performance and potential uses for AbsIDconvert are discussed in the following sections.

4.6 Results

4.6.1 Intervals vs. relational database

The genomic coordinate information for different identifier types mapped to 53 species were stored as intervals. An interval tree method was implemented and used to store and query the corresponding interval information for each identifier type. For comparison with relational databases, an equivalent MySQL database was implemented to perform ID conversion based on coordinate information, and the run time for both of these methods were compared.

Run–time comparisons of the interval tree and MySQL implementations were performed using randomly sampled rat EST IDs which were subsequently converted to Entrez gene IDs. To test the actual runtime, the number of EST IDs was increased exponentially for each test point and the corresponding execution time (in seconds) was measured. The run time complexity of the interval tree maintained a constant rate while the relational methodology grows in linear fashion, allowing for the conversion of millions of identifiers in only a few seconds (Fig. 4.6).

Further analysis of conversion runtime was performed using 1000 random sampled IDs from Affymetrix® Rat230_2 microarray probes, Agilent Cgh105a microarray probes, RefSeq IDs, Ensembl transcripts, Entrez genes, HUGO gene symbols and EST IDs which were converted into one another using the web version of AbsIDconvert (Table 4.3). The extreme left column represents the

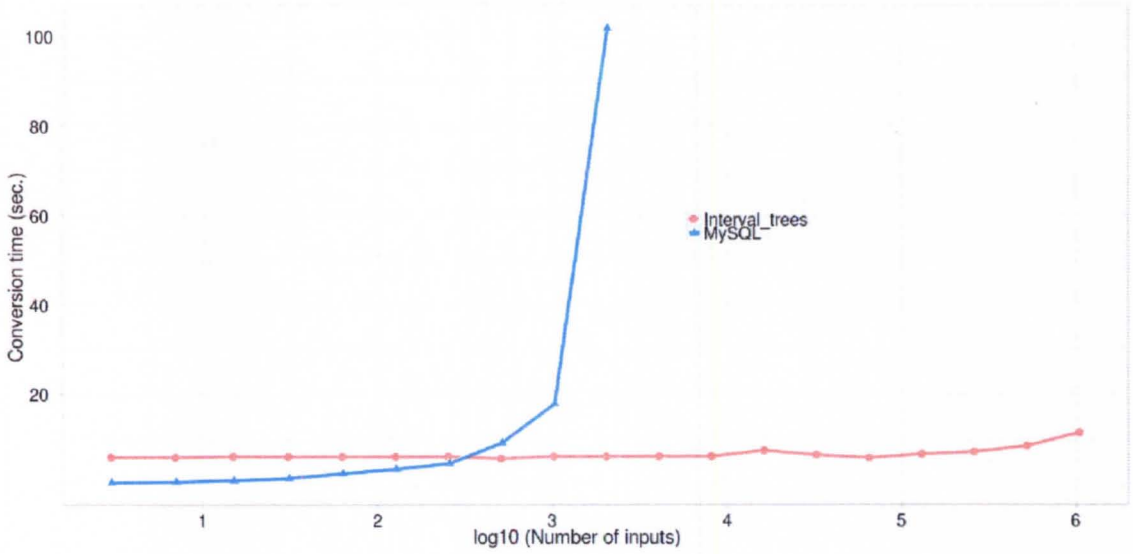

Figure 4.6: Run time comparison between MySQL and interval-trees approach.

source identifiers which are converted to target identifiers shown in first row. The numbers in small parentheses in the first row show the total number of genomic coordinates for individual ID types (For instance, Affymetrix® Rat230_2.0 probes have altogether 231,971 intervals stored). Since AbsIDconvert supports conversion to multiple target types, the last column represents the time elapsed when an input type is converted into all other ID types.

Table 4.3: Run time (sec.) to convert 1000 IDs from one type to another using web–based AbsIDconvert.

	Rat230_2 (231,971)	Cgh105a (97,973)	RefSeq (160,644)	EnsTrans (349,445)	Entrez gene (30,972)	GeneSymbol (30,972)	EST seq (3,918,403)	All
Affymetrix Rat230_2	5.6	3.2	4.1	7.6	3.2	3.3	33	47.6
Agilent Cgh105a	5.1	3.9	2.5	2.7	2.92	3.05	31.3	55.6
RefSeq	4.5	3.1	3.6	3.6	2.3	2.2	31.9	34.5
Ensembl transcript	2.9	3.8	3.1	4	2.47	3.02	34.6	47.1
Entrez gene	2.7	2.9	2.8	3	7.5	7.1	18.4	35.3
Gene symbol	2.9	2.8	2.7	2.9	8.5	7.5	16.6	38.2
EST sequences	18.6	17.6	31	30.3	28.3	29.3	64.1	73.7

4.6.2 Run–time comparison

Direct comparison to other ID conversion approaches is not straightforward due to the differences in annotation information (based on the last available update), supported ID types, and development/deployment platforms. In order to test the runtime of comparable solutions (DAVID,

Clone/Gene ID Converter, GATExplorer, MADGene, and AbsIDconvert), a varying number (100 to 30,000) of Affymetrix® Rat230_2.0 microarray probesets were converted to Entrez IDs (Fig. 4.7). When the number of probes sets converted was small (100), the conversion time for all tools was nominal. For a moderate number of probe sets (5,000) MADGene, DAVID and AbsIDconvert performed similarly (12.6, 6.1 and 5.1 sec. respectively), while GATExplorer took around a minute and Clone/Gene ID Converter took 15 minutes (Fig.4.7(a)). As the number of probe sets further increased, all of the tools, with the exception of MADGene and AbsIDconvert, were incapable of tractably handling such a large number of inputs. Since the Affymetrix® Rat230_2.0 has roughly 31,000 unique probe sets and over 300,000 individual perfect match probes, a run time comparison for a large number of inputs ($> 30,000$) was performed by converting randomly sampled human transcripts into Entrez IDs (direct conversion of individual probes is not possible within all of the tools; therefore the closest comparison is made to the same number of human transcripts). For 100,000 inputs, only MADGene and AbsIDconvert completed successfully, taking 45 sec and 24 sec, respectively (Fig.4.7(b)). Note that DAVID limits user input to 30,000 identifiers. The run–time complexity for AbsIDconvert compares favorably to other similar tools, demonstrating its applicability in the analysis of high throughput data.

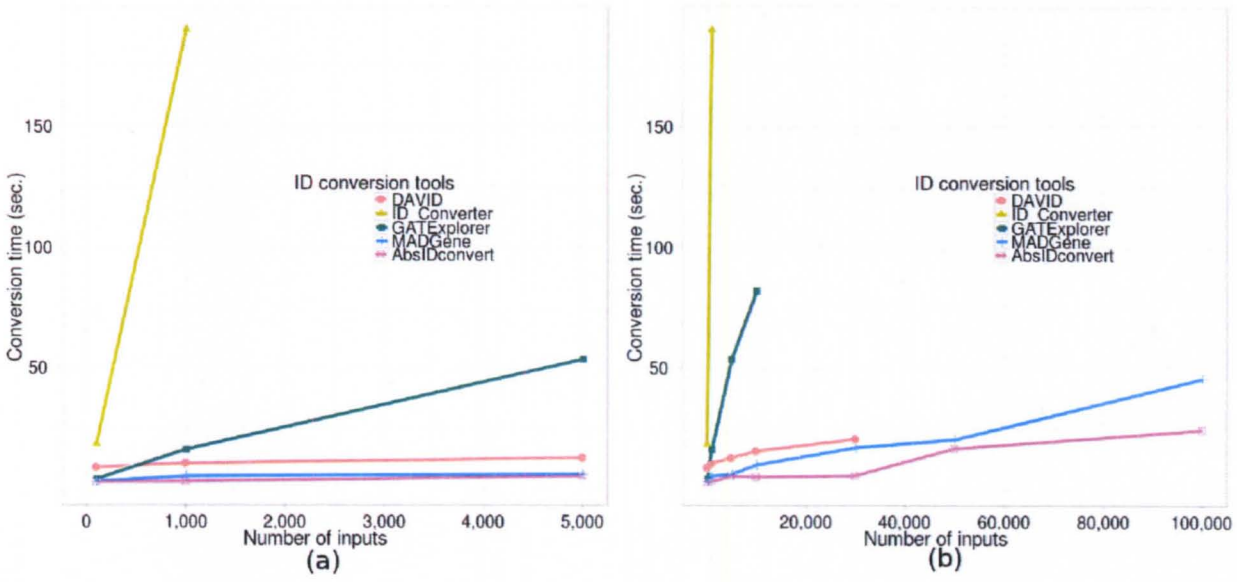

Figure 4.7: Run time comparison for ID conversion.

4.6.3 Output accuracy

The accuracy of conversions performed using AbsIDconvert was assessed based on the overlap of the successfully converted IDs with those found using other tools for three types of conversions. In the first conversion, 1000 unique Entrez IDs were randomly sampled from the "org.Hs.eg.db" Bioconductor annotation package and converted to their corresponding official gene symbols. Ten ID conversion tools, from a total of 19 tools listed in Table 4.1, can perform this conversion. Considering NCBI as the authority for Entrez IDs, the accuracy of different conversion tools were evaluated using the following assumptions:

1. NCBI contains the most up to date information and its annotations are correct.

2. An Entrez ID may be annotated by more than one gene symbol.

3. Given an Entrez ID x, if a tool converts x to a set of gene symbols, Y ($x \rightarrow Y$), and NCBI annotates x to another set of gene symbols, Z ($x \rightarrow Z$), then accuracy terms can be defined as:

   - **True positives (TP)** are those conversions in which the converted gene symbol set contains all the gene symbol(s) annotated by NCBI (i.e. $Z \subseteq Y$).
   - **False positives (FP)** are unexpected results. This includes incorrect conversions ($Z \nsubseteq Y$), as well as those conversions in which NCBI does not annotate an Entrez ID with any gene symbol, but a tool finds some gene symbol corresponding to that Entrez ID ($Z = \phi$ and $Y \neq \phi$).
   - **False negatives (FN)** are missing conversions in which a tool could not find corresponding gene symbol(s) ($Z \neq \phi$ and $Y = \phi$).
   - **True negatives (TN)** are the correct absence of conversion in which NCBI as well as a particular tool does not convert an Entrez to any gene symbol ($Z \neq \phi$ and $Y \neq \phi$).

4. Accuracy is defined as

$$\%Accuracy\ (ACC) = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

Table 4.4 shows the contingency table and associated statistics for the conversion of 1000 Entrez IDs to gene symbols. AbsIDconvert converted a total of 885 Entrez IDs with an accuracy of 87.2% followed by DAVID (853, 79.1%), MADGene (854, 73.1%) and HMS & IC (724, 72.9%). Although Onto-Translate converted a total of 823 Entrez IDs, it has more FP conversions than HMS & IC and therefore a lower accuracy. We further investigated the conversions from the top four tools on the basis of their accuracy and summarized the results in a Venn diagram (Fig. 4.8(a)). AbsIDconvert converted a total of 83 Entrez IDs which are missed by the other tools. NCBI places all these Entrez IDs onto the reference genome and annotates them with gene symbols that are in agreement with AbsIDconvert (Table A.1). Of these 83 Entrez IDs, 48 are categorized as "pseudo", 27 as "miscRNA", four as "protein-coding", three as "unknown" and one as "other". AbsIDconvert was unable to convert a total of 115 Entrez IDs, out of which 21 IDs were not converted by any of the tools examined.

Table 4.4: Entrez ID to gene symbol conversion accuracy.

Tool	totalMapped	TP	FP	FN	TN	TPR	FPR	**ACC**	FDR	F1_score
AbsIDconvert	885	866	19	109	6	88.82	76.00	**87.20**	2.15	93.12
DAVID	853	790	63	146	1	84.40	98.44	**79.10**	7.39	88.32
MADGene	854	730	124	145	1	83.43	99.20	**73.10**	14.52	84.44
HMS & IC	724	723	1	270	6	72.81	14.29	**72.90**	0.14	84.22
Onto-Translate	823	722	101	176	1	80.40	99.02	**72.30**	12.27	83.90
MatchMiner	539	457	82	458	3	49.95	96.47	**46.00**	15.21	62.86
Clone/Gene ID converter	537	441	96	457	6	49.11	94.12	**44.70**	17.88	61.46
g:Convert	445	433	12	549	6	44.09	66.67	**43.90**	2.70	60.69
Synergizer	445	433	12	549	6	44.09	66.67	**43.90**	2.70	60.69
Babelomics	486	421	65	508	6	45.32	91.55	**42.70**	13.37	59.51

Of the 94 Entrez IDs that AbsIDconvert was not able to convert but other tools were (Table A.2), most were either "not on current assembly", meaning that the reference sequence for that Entrez ID could not be mapped to the current genome (28 IDs), but could be mapped to previous genome assemblies; or "not annotated on reference assembly", indicating that the sequence cannot be found on the reference assembly at all (61 IDs). Five conversions were found where the Entrez IDs reported had since been deleted and replaced (DAVID and MADGene both converted these IDs).

In a second conversion test, 1000 randomly sampled Entrez IDs were converted to RefSeq IDs using ten of the 19 tools listed in Table 4.1 (the others are not able to perform this type of conversion and were not evaluated). There are many different classes of RefSeq IDs, including mRNA (ID starts with NM_), RNA (NR_), protein (NP_), as well as predicted versions of each one (XM_ , XR_ and XP_ respectively). How RefSeq IDs are segregated for conversion differs among the tools tested. For example, a number of tools combine all the different types of RefSeq IDs into one converted ID type while others treat each one separately. Other tools ignore the predicted RefSeq IDs and only consider mRNA and RNA. For example, AbsIDconverts RefSeq database combines both mRNA and RNA, whereas MADGene includes predicted products (XM). DAVID and Synergizer have separate options for RNA and mRNA RefSeq. Therefore, to enable comparison between all the tools, only those conversions that result in mRNA or RNA RefSeq IDs are considered, and for those tools that report them separately, the results from both conversions were combined. In addition, any predicted RefSeq IDs (i.e. those that begin with X) were removed.

Using the same assumptions as reported for the Entrez to Symbol conversion, the accuracy of conversion for each tool was calculated (Table 4.4). Of the 1000 Entrez IDs used, NCBI annotates only 599 with one or more RefSeq. In this case, the accuracy for the various tools ranged from a high of 75.6% (AbsIDconvert) to a low of 38.9% (HMS & ID).

Table 4.5: Entrez ID to RefSeq conversion accuracy.

Tool	Total Mapped	TP	FP	FN	TN	TPR	FPR	**ACC**	FDR	F1_score
AbsIDconvert	586	362	224	20	394	94.76	36.25	**75.60**	38.23	74.79
MADGene	551	335	216	49	400	87.24	35.06	**73.50**	39.20	71.66
Onto-Translate	501	291	210	99	400	74.62	34.43	**69.10**	41.92	65.32
DAVID	549	311	238	72	379	81.20	38.57	**69.00**	43.35	66.74
Synergizer	482	278	204	121	397	69.67	33.94	**67.50**	42.32	63.11
g:Convert	482	278	204	121	397	69.67	33.94	**67.50**	42.32	63.11
MatchMiner	474	268	206	126	400	68.02	33.99	**66.80**	43.46	61.75
Babelomics	501	267	234	128	371	67.59	38.68	**63.80**	46.71	59.60
Clone/Gene ID converter	421	219	202	195	384	52.90	34.47	**60.30**	47.98	52.46
HMS & ID	461	227	430	181	162	55.64	72.64	**38.90**	65.45	42.63

The results from the four most accurate tools were investigated further. 497 Entrez IDs were converted commonly by all tools (Fig. 4.8(b)). AbsIDconvert converted 586, followed by MADGene (551), DAVID (549) and Onto-Translate (501). Five conversions specific to MADGene were not

found by AbsIDconvert (Table A.3). In this case, AbsIDconvert correctly mapped the Entrez IDs to the genome (Table A.4); however, the corresponding RefSeq IDs were not in the data obtained from UCSC. Other conversions that AbsIDconvert did not report were found to be false positives reported by other tools. For example, DAVID and Onto-Translate both reported converting "4586" to "NM_017511" and "441956" to "NM_001013729"; however, the genomic intervals for those IDs do not overlap, and both RefSeq IDs are shown in NCBI as "permanently suppressed". For the twenty conversions specific to DAVID, the reported RefSeq IDs were found to be associated with different Entrez IDs in NCBI (Table A.5).

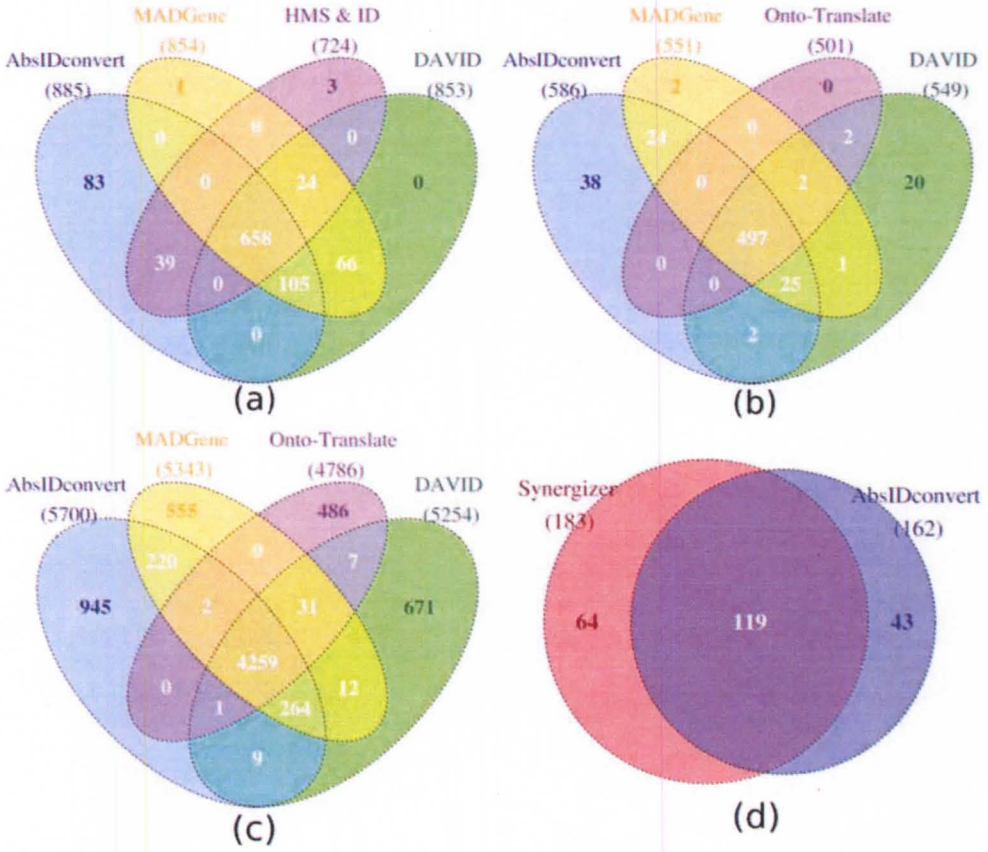

Figure 4.8: Venn diagram showing the conversion results.

The thirty-eight Entrez IDs converted only by AbsIDconvert were investigated further to verify whether they were "correct". Thirty-three are in agreement with the NCBI data (Table A.6). For the other five, we examined the genomic intervals of both the Entrez IDs and reported RefSeq IDs to verify that they do indeed overlap (intervals are reported in Table A.7). In all cases the converted

IDs do have overlapping intervals with two of the Entrez IDs discontinued and replaced since the initial construction of the AbsIDconvert database, "100505905" (to "23189" on March 2, 2012) and "100652874" (to "100505641" on Feb 3, 2012).

To better assess the accuracy of AbsIDconvert compared to other tools, the Entrez to RefSeq ID conversion was repeated ten times, randomly choosing 1000 Entrez IDs each time. Out of the 10,000 randomly selected Entrez IDs, 8,974 were unique. AbsIDconvert converted 5700 (63%), followed by MADGene (5343, 59.5%), DAVID (5254, 58.5%) and Onto-Translate (4786, 53.3%) (Fig. 4.8(c)). A total of 945 (10%) of the IDs were exclusively converted by AbsIDconvert.

In the third conversion, 1000 randomly sampled human Affymetrix® GeneChip HG-U133 Plus 2.0 probesets were converted to Agilent Cgh44b probes (Fig 4.8(d)). This type of cross-platform conversion is important in meta-analysis studies where results are drawn by integrating and analyzing data from a number of independent studies/platforms. As this type of conversion is available only in Synergizer, we compared the conversion results of this tool with AbsIDconvert. Synergizer converted 183 whereas AbsIDconvert converted 162 probesets. The reason for the small number of conversions is primarily due to the design differences of the probes on these chips. Two questions required deeper investigation: 1. Why was AbsIDconvert not able to convert 64 Affymetrix® IDs that were successfully converted by Synergizer; and 2. Are the 43 conversions exclusive to AbsIDconvert valid? To answer these, we extracted the design annotation of all the Affymetrix® GeneChip HG-U133 Plus 2.0 probesets provided by Affymetrix's NetAffx [148] along with the design annotations for the Agilent Cgh44b probes supplied by Agilent [149]. These provided the individual locations of each probe on the hg19 genome, thereby enabling investigation of the interval separation between the probesets.

In order to examine the 64 probesets converted by Synergizer but not by AbsIDconvert, the genomic location(s) of the Affymetrix® probesets were compared to the genomic locations of the Agilent probes. Fifty-six (out of 64) of the probes are separated according to their genomic locations and do not overlap at all. This separation ranges from 75 to 418,671 BP with a median separation

of 4,736 bases. Further analysis determines that these all lie in the regions between the individual probes of the respective probesets and therefore have no shared sequence identity.

Most of the ID converter tools including Synergizer map the genetic entities (probes, probesets) spanning tens of bases to an intermediary such as Ensembl that is at a coarser granularity spanning a few kilobases with possible intronic regions. While performing conversions, these tools only use the probe annotation, disregarding the actual sequence information. The above false positives provided by Synergizer are likely the result of ignoring the sequence level information as the two types of probes actually span different genomic intervals.

Next we considered conversions found exclusively by AbsIDconvert. Based on the official annotation from NetAffx™, we found that intervals for all 43 Affymetrix® probesets actually contain or overlap the converted Agilent probes with a mean overlap of 56.43 bases. Considering that most of the Agilent probes are 60 bases long and an Affymetrix® probeset contains overlapping 25 bp probes, this indicates most of these Agilent probes are contained in the Affymetrix® probeset region. These probesets were checked at the probe level and it was determined that these converted Agilent probes overlap with individual Affymetrix® probes to some extent, or are completely contained with a mean overlap length of 38.70 BP. We are not sure why Synergizer was unable to convert these 43 probes; however, the official annotation confirms these annotations and bolsters our confidence in the power and accuracy of our sequence based ID conversion.

4.7 Case studies

Three illustrative case studies were explored to demonstrate the capabilities of AbsIDconvert. The first case study considers sequence-based mapping of identifiers in a comparative genomics analysis of organisms involved in malaria; the second examines remapping of probes to annotations within and across species using a historical cDNA platform from Incyte; and the third identifies Ensembl transcripts mapped by Agilent and Affymetrix® arrays.

4.7.1 Case study 1: Comparative genomics: plasmodium mapped to human and *Anopheles gambiae*

Recent studies have surveyed the role of both host and pathogen genetic variability to determine molecular signatures for host-pathogen interactions [150]. While the interactions between a pathogen and its host are often mediated by the host immune system responses to the pathogen, host-pathogen relationships theoretically have the potential to create a metagenomic environment whereby the total transcriptome is contributed by both the host and pathogen genes. In some cases, such as *Neisseria meningitidis*, a direct interaction between host and pathogen genes has been demonstrated [151]. As an illustrative example, it might be possible that shared sequence similarities between pathogen and host genes play a role in host gene regulation via pathogen genes and gene products that provide additional promoter sites, miRNA targets, and binding motifs similar to those found in the host. To test the feasibility of this possibility in the context of malaria, we used absIDConvert to identify coding sequences identical between the PF and PV species and the human and anopheles genomes.

Plasmodium is a parasite responsible for causing malaria in humans primarily in tropical and sub–tropical areas. About 3.3 billion people are at risk of this disease, leading to 250 million malaria cases and one million deaths worldwide every year (`http://www.who.int/features/factfiles/malaria/`). Altogether four Plasmodium species are responsible which are carried by the female *Anopheles gambiae* mosquito. *Plasmodium falciparum (PF)* and *Plasmodium vivax (PV)* are the most common, with PF being the deadliest.

Coding sequences for each gene for these two species were downloaded from the PlasmoDB website (`http://plasmodb.org/`) [152]. The total number of coding sequences in PF and PV were 5,524 and 5,435 respectively. Sequences for each of these genes were then fragmented into 50 base-pair (BP) long sequences with an overlap of 25 BP. The fragmented sequences were given a unique name by attaching a numerical suffix onto the gene name that denotes the order of appearance in the gene sequence. These fragmented sequences were analyzed using AbsIDconvert by selecting default

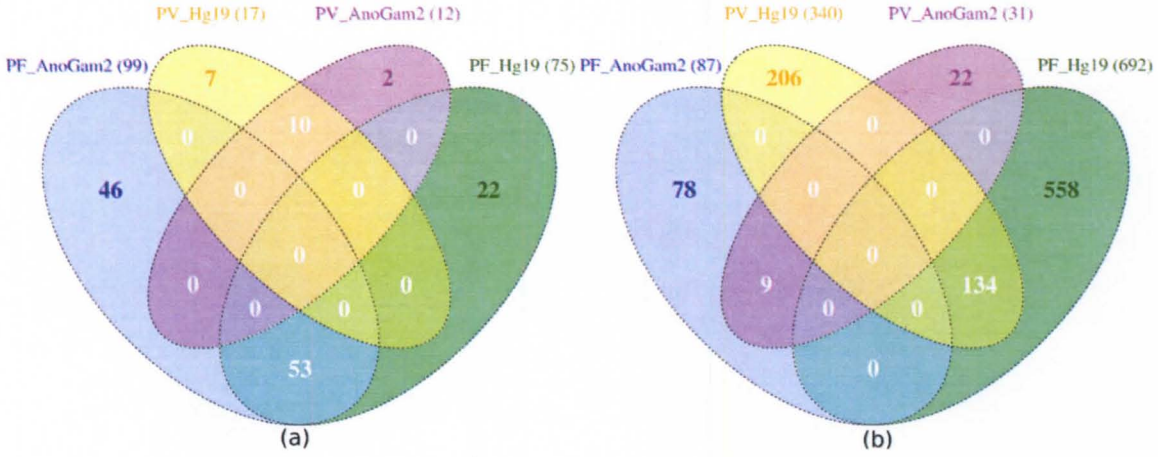

Figure 4.9: (a). Number of gene fragments from PF and PV that overlaps with at least one gene from *Anopheles gambiae* and *Homo sapiens*. (b). Corresponding genes in *Anopheles gambiae* (AnoGam2) and *Homo sapiens* (hg19) that were mapped by gene fragments from PF and PV.

parameters including no mismatch while aligning to the *Anopheles gambiae* (AnoGam2) and *Homo sapiens* (hg19) genomes (Fig. 4.9).

A total of 75 gene fragments from PF (PF_Hg19 in Fig. 4.9(a)) had an exact sequence match to 692 human genes (PF_Hg19 in Fig. 4.9(b)). For PV, the aligned number of gene fragments and corresponding genes were 17 (PV_Hg19 in Fig. 4.9(a)) and 340 (PV_Hg19 in Fig. 4.9(b)), respectively. These numbers indicate that the gene fragments align to multiple locations on the human genome. Among genes that were mapped from PF and PV gene fragments, a total of 134 genes were common. When the same gene fragment sequences from PF and PV were aligned to the *Anopheles gambiae* genome (AnoGam2), a total of 99 (PF_AnoGam2 in Fig. 4.9(a)) gene fragments from PF were mapped to 87 (PF_AnoGam2 in Fig. 4.9(b)) different genes, showing that the correspondence between the gene fragments and genes is largely one–to–one. These numbers for PV were 12 (PV_AnoGam2 in Fig. 4.9(a)) and 31 (PV_AnoGam2 in Fig. 4.9(b)), respectively.

A more detailed analysis of the genes identified using ontological information indicates a significant enrichment in cell adhesion processes (Table 4.6). These are present in the GO terms 'cell-cell adhesion' (and others), but also implied by the large number of terms regarding neuronal axonogenesis and synapse formation, which require specific regulation of cellular adhesion. While purely

speculative at this point, it is possible these plasmodium genes interact with the human host to help sequester human erythrocytes in small blood vessels which aids in the invasion plasmodium into the immune system [153]. While benchtop analysis of these genes is needed to determine if the "feasible" actually occurs, it is clear that analysis using AbsIDconvert has identified, via cross-species analysis, a limited set of genes that can be further interrogated for understanding the malaria-related pathophysiology, including the process of plasmodium incorporation into erythrocytes.

Table 4.6: Significantly enriched (p-value < 0.001, number of genes ≥ 2) Gene Ontology biological processes for the *P. falciparum* and *P. vivax* genes.

GO ID	Description	listMembership	pFal.Pvalue	pViv.Pvalue
GO:0048639	positive regulation of developmental growth	pFal	0.00023	0.078421
GO:0051865	protein autoubiquitination	pFal	0.000611	0.310842
GO:0007417	central nervous system development	pFal	0.000749	0.052751
GO:0010559	regulation of glycoprotein biosynthetic process	pFal	0.000534	0.189699
GO:0043062	extracellular structure organization	pFal	0.000896	0.056366
GO:0031290	retinal ganglion cell axon guidance	pFal	0.000729	0.020543
GO:0050772	positive regulation of axonogenesis	pFal	0.000671	0.108078
GO:0007268	synaptic transmission	pFal	9.63E-005	0.004437
GO:0007156	homophilic cell adhesion	pFal	2.90E-005	0.00181
GO:0048745	smooth muscle tissue development	pFal	0.00097	0.215514
GO:0008038	neuron recognition	pFal,pViv	0.000611	2.71E-005
GO:0071702	organic substance transport	pViv	0.358064	0.000932
GO:0010827	regulation of glucose transport	pViv	0.15634	0.000705
GO:0016337	cell-cell adhesion	pViv	0.002316	0.000615
GO:0045725	positive regulation of glycogen biosynthetic process	pViv	0.316458	0.000806
GO:0008037	cell recognition	pViv	0.041274	0.000425
GO:0010907	positive regulation of glucose metabolic process	pViv	0.486254	0.000312
GO:0045913	positive regulation of carbohydrate metabolic process	pViv	0.561654	0.000731
GO:0010676	positive regulation of cellular carbohydrate metabolic process	pViv	0.561654	0.000731
GO:0030036	actin cytoskeleton organization	pViv	0.133792	8.55E-005
GO:0030029	actin filament-based process	pViv	0.099308	2.74E-005

4.7.2 Case study 2: Reinterpretation of prior datasets

Annotations used for DNA microarray studies quickly become out–of–date as more knowledge emerges about a species' transcriptome. In addition, there are instances where one microarray platform may be used to measure gene products from a comparative species. For example, Incyte arrays spotted with human ESTs have been used to query gene expression levels in mouse and/or rat, based on the assumption that the human ESTs would bind to and provide measurements of the corresponding gene in rodents [154–156]. Using the original EST sequences spotted on the array from these studies, we sought to verify the current annotations of the ESTs, and also

determine which rodent genes should bind the ESTs based on sequence alignment to the human, mouse, and rat genomes. Original EST sequences were found by searching two sources using the Incyte IDs supplied on the chip. The first source was the NCBI EST database, using a search string composed of IMAGE: and the Incyte clone ID number (identifies clones generated from the IMAGE consortium sequencing project). The second source was the Open Biosystems database (`http://www.openbiosystems.com/`), using a search string composed of LIFESEQ and the clone ID number. In some instances, multiple EST sequences were returned for each clone ID. A total of 8,392 sequences were downloaded and aligned to the genomes of human, rat, and mouse using AbsIDconvert with the default BLAT settings. The genome wide best alignment was found for each probe by considering only those alignments falling within 5% of the maximal alignment score (Fig. 4.10(a)). Corresponding to each of these aligned coordinates, overlapping Entrez IDs were found for all three organisms. Out of the 7,095 human Incyte IDs which had corresponding genomic interval(s), 4,155 have at least one human Entrez ID associated with them. This number was 2,081 (out of 3,368) for mouse and 1,438 (out of 2,776) for rat (Fig. 4.10(b)).

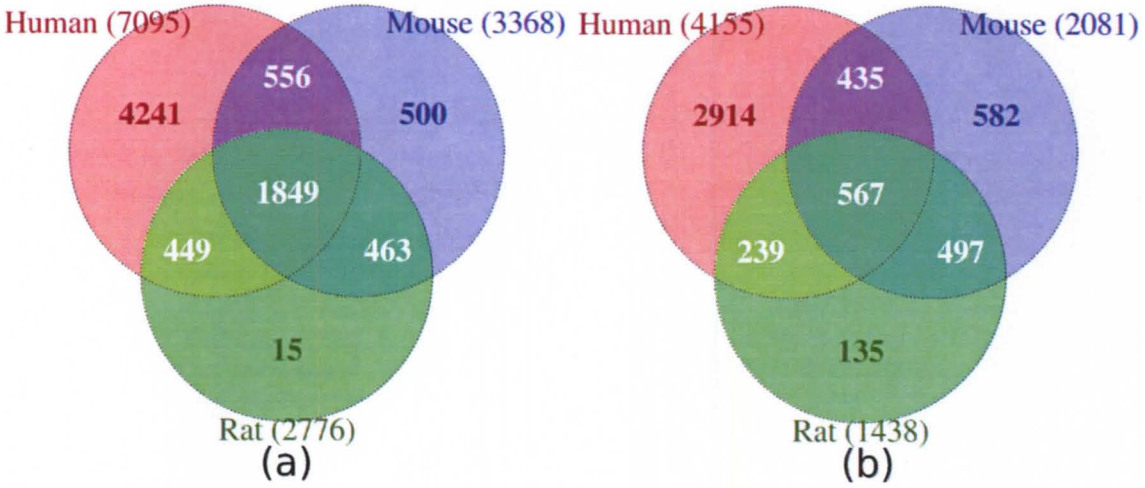

Figure 4.10: (a). Number of Incyte IDs mapping to the human, mouse and rat genomes within 5% of the maximum alignment score. (b). Incyte IDs with at least one Entrez ID found using AbsIDconvert.

Homologous genes can be compared across species using NCBI's Homologene resource [157] when gene names are known. However, if sequence information is available, it would be best to use that

sequence information to determine if homology exists based on sequence conservation, particularly in cases where probes of known sequence are being used to measure a specific gene, such as in DNA microarrays or in–situ hybridization. Both methodologies were applied to the Incyte array used in [154–156].

For the Homologene based comparison, all of the Incyte IDs that map to at least one Entrez ID using AbsIDconvert were used to determine if a homologous gene exists, and if so, if there are corresponding entries for each of the species studied. Similarly, for those Incyte probes matching at least one Entrez ID, the sequence was used as a query into each of the other species using AbsIDconvert to determine if the probe maps to and overlaps an Entrez ID in a cross-species sense. As Table 4.7 indicates, using the Homologene conversion alone yields a high number of homologs (82% – 88%); however, using the sequence level information, it can be seen that a much lower percentage of probes (19% – 74%) actually map to known Entrez gene regions in the other species. These demonstrate that only a small number of the probes on the array should be utilized for cross species comparisons.

4.7.3 Case study 3: Meta–analytic studies across platforms

Meta-analysis enables the integration of many different experiments with a common research hypothesis. However, high-throughput -omics meta-analyses are hindered due to the heterogeneity of DNA microarray array designs (length and location of probes), data acquisition, analysis, and inter- and intra-study variability. Therefore, many meta-analyses use the same species or even the same array platform to mitigate some of these heterogeneities. However, many studies do still attempt to perform cross-platform and inter-species meta-analyses, and tools such as AILUN (Array Information Library Universal Navigator) [158], A-MADMAN (Annotation-based microarray data meta-analysis tool) [159], and LOLA (List Of Lists Annotated) [160] enable cross-species meta-analysis using Entrez ID, gene symbol or other IDs as a conversion intermediary. AbsIDconvert can perform cross-platform / -species analysis efficiently using the sequence based approach. We

Table 4.7: Comparison of Homologene and sequence based homologs.

Organism	mapped†	Entrez ‡	Homol§	Human (Hom)	Mouse (Hom)	Rat (Hom)	Human (Seq)	Mouse (Seq)	Rat (Seq)
Human	7095	4155	3854	–	3648 (88%)	3401 (82%)	–	1002 (24%)	806 (19%)
Mouse	3368	2081	1872	1794 (86%)	–	1715 (82%)	1002 (48%)	–	1064 (51%)
Rat	2776	1438	1263	1210 (84%)	1222 (85%)	–	806 (56%)	1064 (74%)	–

mapped†:Number of probes mapped to Genome; Entrez‡:Mapped probes with Entrez ID; Homol§: Probes with Entrez ID as well as Homologene ID; Hom: Homologene Based Homologs; Seq: Sequence Based Homologs determined using AbsIDconvert.

previously demonstrated that AbsIDconvert efficiently and accurately converted Affymetrix® HG_-U133Plus2.0 probes into Agilent Cgh105a probes, among other types of conversions.

To determine how comparable two microarray studies using different array platforms on a common organism could be, Affymetrix® HG_U133Plus2.0 and Agilent Cgh105a probe sequences were mapped and converted to corresponding human Ensembl transcripts using the default AbsIDconvert parameters. For the Affymetrix platform, 423,815 out of 603,158 probes were mapped to one or more transcripts, with 94,713 of the total Ensembl transcripts (173,742) being mapped (Fig. 11). This leaves 79,029 Ensembl transcripts that were not mapped by any Affymetrix® probes. For Agilent, 27,184 (out of 99,026) mapped to 60,829 Ensembl transcripts. 79,029 (45% of the total) Ensembl transcripts do not have any mapped Agilent Cgh105a probes. The number of shared Ensembl transcripts between platforms was surprisingly small (46,308), indicating that each platform appears to have probe specific subsets of Ensembl transcripts. The number of Ensembl transcripts not probed by either platform was surprisingly large. This appears to be due to a lack of probes designed to bind those Ensembl transcripts, as the majority of unmapped transcripts are much shorter than those that are mapped (Fig. 4.12). As Fig. 4.11 illustrates, 46,308 transcripts should be directly comparable between Affymetrix® HG_U133Plus2.0 and Agilent Cgh105a, while a large number of transcripts are not available in one or the other (or both) platforms.

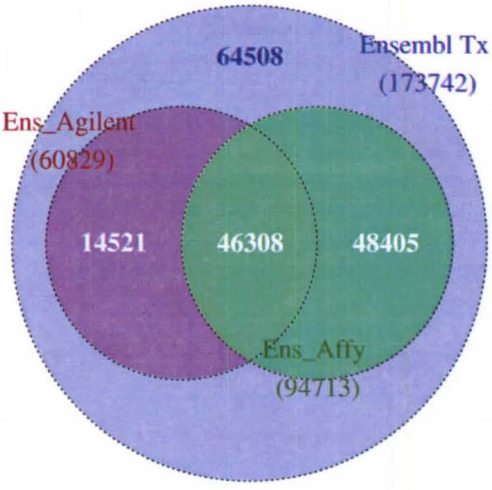

Figure 4.11: Ensembl transcripts mapped by Agilent Cgh 105a and Affymetrix® HG_U133Plus2.0 probes.

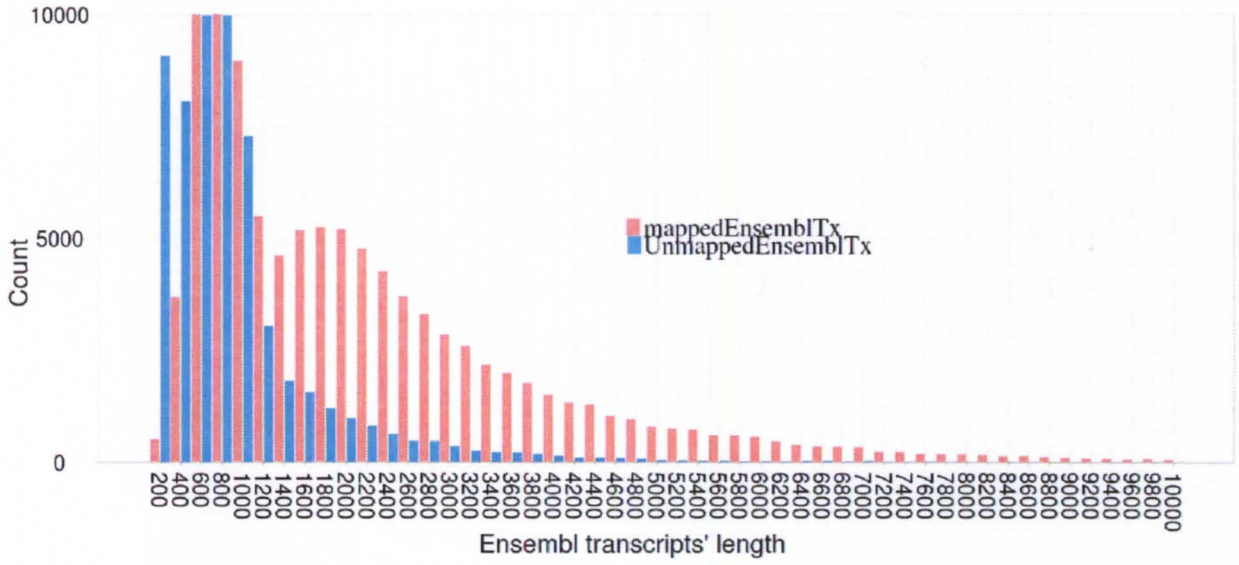

Figure 4.12: Exonic lengths of Ensembl transcripts mapped/ unmapped by probes

4.8 Conclusion

AbsIDconvert is the only known gene ID conversion tool based on genomic coordinates / intervals of which we are aware. This is a novel and important contribution in the realm of gene ID conversion due to the large variety of genetic entities in current use by biologists, the need to convert between them, and the fact that most biological entities (nucleic acid, protein entities etc.) have an associated sequence. Mapping of the entity sequence to a reference genome sequence provides the concomitant genomic interval that allows determination of other entities that have overlapping genomic intervals.

The interval basis of AbsIDconvert provides ease of flexibility with respect to any additions, deletions or updates of the underlying objects, requiring only adding of intervals, removing intervals, or modifying the intervals themselves, respectively. This makes it possible to easily keep the structure updated as the current state of biological knowledge changes. A major update is only required when the underlying genome changes, a fairly rare occurrence for most organisms, especially when compared to how often other genomic databases are modified.

These intervals also allow easy discovery of genetic entities that only partially overlap with queried IDs / intervals, or that are within a specified distance nearby. More frequently, researchers are interested in those genes that are near specific genomic intervals corresponding to various types of

genetic control elements such as transcription factor binding sites, enhancers, untranslated regions, and hyper / hypo methylated regions. AbsIDconvert makes it easy to find those entities that overlap or lie nearby regions of interest. With the incorporation of a sequence mapping algorithm, AbsIDconvert integrates the determination of genomic intervals for any supplied sequence, making it possible to easily find and convert between IDs from any platform and organism, such as the examination of correspondence of the human EST clones with rat and mouse genes (case study 2) and of plasmodium and human genes (case study 1). We do not know of any other system that can easily accomplish these types of analyses.

AbsIDconvert can greatly facilitate the work of those who are involved in meta analyses studies. When comparing studies where either the species and / or platform varies, this methodology will have clear advantages over others as it is based on common genomic coordinates.

The use of an interval tree structure makes it possible to perform large conversions quickly and efficiently. This method is efficient while dealing with genomic intervals and has a significant advantage over other methods such as relational databases. Although theoretically limited by working memory, none of the interval trees generated and used by AbsIDconvert require more than 300MB of RAM on the deployed server, with the majority being rather small in size (less than 10 MB). If the data cannot fit into main memory, a method such as that proposed by Arge et al. [102] [103] can be used that maintains the interval tree in secondary memory efficiently.

AbsIDconvert is provided as a web page at `http://bioinformatics.louisville.edu/abid/`, and is also available as a virtual machine for those wishing to run a local instance. Future work will include providing command line access, a RESTful interface, and modifying the interface to utilize a workflow management tool for genomic data such as GALAXY, where the primary data units are genomic sequences and intervals.

CHAPTER 5

A HEURISTIC ALGORITHM FOR DETECTING INTERCELLULAR INTERACTIONS

5.1 Introduction

Cell–cell interactions are important aspects of many biological processes. Examples include migratory processes (e.g., immune cell transvascular migration, nervous system development, and cancer metastasis), binding processes (e.g., oocyte implantation and leukocyte tethering and rolling), induction processes (e.g., stem cell generation and floor–plate or roof–plate modulation of neuronal fate), and adaptation/plasticity processes (e.g., neovascularization, axonal regeneration or sprouting, and sequestration of cancerous or infected cells). Chemical factors released by the skin both constitutively and in response to various stimuli activate receptors expressed by the sensory axons providing innervation of the skin [161]. This activation can initiate a signaling process which ultimately influences neuronal structure and/or function by affecting transcription and translation. The structural nature of the nervous system is unique in that for a single cell the location of the initiation of the signaling cascade (skin) and the location of transcription/translation (sensory neuron cell body in the dorsal root ganglia between vertebrae) can be separated by great distances, in cases of large animals up to many meters. Thus, the interaction site and the transcription/translation site represent different tissue samples, and are run separately for proteomic and transcriptomic analyses. Information regarding intercellular interactions, particularly when the interacting elements are represented in separate samples, is generally not efficiently/accurately extracted with existing analytical tools, but may be extracted by examining the list of regulated genes/proteins against databases of known molecular interactions.

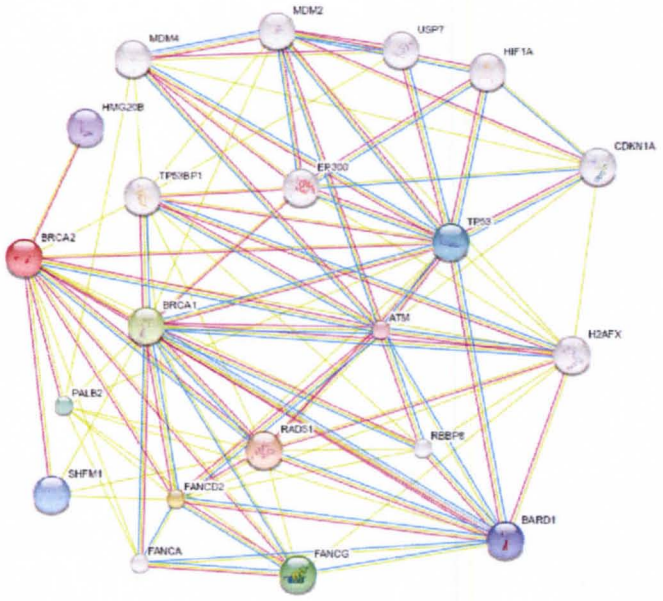

Figure 5.1: Evidence view of BRCA2 protein interactions from the STRING protein database.

Given a list of relevant genes or proteins (here, the concept of gene and protein is used interchangeably) reacting to a given condition, it is a simple process to find all interactions within the gene list and with other genes using known interaction network data. As an example, Fig. 5.1 shows the interaction network of BRCA2 from the STRING [162] database. BRCA2 is connected to many genes or proteins. These interactions may be direct or indirect through a transitive relationship. For example, BRCA2 is connected directly to BRCA1 and indirectly to ATM via BRCA1. As the number of genes in the list increases, the complexity of the network generated will also increase exponentially.

It has been widely established that cell–cell interactions are mediated via protein–protein (gene–gene) interactions. Having lists of genes that are differentially expressed from two different tissues, it is of interest to determine how the expression of genes in one tissue might influence gene expression in another tissue. The influence may be positively correlated (up–regulation of gene A in tissue 1 up–regulates gene B in tissue 2) or negatively correlated (up–regulation of gene A in tissue 1 down–regulates gene B in tissue 2). The signal may be carried from one tissue to another via a number of intermediate proteins. Therefore, it would be advantageous to find all possible interactions between

two sets of genes with up to n–intermediaries. Finding all possible paths leading to these interactions is computationally intensive especially when the number of interactions is on the order of hundreds of thousands as each of the nodes may interact with hundreds of other nodes.

To solve this problem, a heuristic method is developed that combines the "Backtracking Algorithm" and a novel concept of exclusion vector (EV). The EV supports two functions: (1) restricting the interaction search space at each iteration; (2) restricting the search space based on defined properties of the proteins. In this work, the location of the proteins according to the cellular component annotation in the Gene Ontology (GO) [82] is used. This method can be readily applied to separate tissue samples that interact, such as neuronal cell bodies and their target tissues, or specific cell-types separated from their native tissue (for example, laser-capture or FACS).

5.2 Interaction databases

To find the pathways or interactions in which a particular gene is involved, we need to search into the available interaction databases. There are many interaction databases publicly available such as PathwayCommons (`www.pathwaycommons.org`), STRING [162], STITCH [163] [164], HaPPI [165], InPrePPI [166], KEGG [111], BioCarta, GenMapp [167], BioGRID [168], MINT [169] and IntAct [170]. A detailed review pertaining to the protein–protein interactions and pathway databases and visualization software can be found in [171].

PathwayCommons provides a common platform to access pathway information from multiple sources represented in a common format. It collects, stores and integrates pathway and interaction information from various publicly available databases. These interaction include biochemical reactions, complex assembly, transport and catalysis events, and physical interactions involving proteins, DNA, RNA, small molecules and complexes. As of February 2012, it contains 442,182 interactions, 1,668 pathways and 86,282 physical entities spanning across 414 organisms. This database can also be accessed programmatically via a web service API. For example, the command *get_pathways* retrieves all pathways involving a particular physical entity such as BRCA1. PathwayCommons import data from the databases which store the interaction data in BioPAX format (`http://www.biopax.org/`).

BioPAX is a common standard format to enable integration, exchange, visualization and analysis of biological pathway data.

STITCH (Search Tools for Interaction of Chemicals) integrates data from different sources such as bench–top experiments, databases and literature to mine known and predicted interactions of chemicals and proteins. The scoring method adopted gives more weight to manually curated interaction while a relevance score is attached to the interactions that are based on experimental information. To search for interactions in chemical databases, STITCH uses the SMILES (Simplified Molecular Input Line Entry System) (`www.daylight.com`) strings and the InChI(IUPAC's International Chemical Identifier) codes. As of May 31^{st}, 2012, STITCH contains interaction for over 300,000 small molecules and over 2.6 million proteins in 1,133 organisms (`http://stitch.embl.de/`). The interaction database as well as the query results of interactions are publicly available to download from `http://stitch.embl.de/`. STRING (Search Tool for the Retrieval of Interacting Genes/proteins) is a similar database of physical and functional interaction of proteins. It relies upon the manually curated data from primary interaction databases such as BioGRID, IntAct, MINT, and BIND and combines it with the information extracted from pathway databases such as KEGG, EcoCyc and Reactome. STRING also incorporates protein–protein prediction algorithms. The database currently conains 5,214,234 proteins across 1133 organisms (`http://string.embl.de/`).

HAPPI (Human Annotated and Predicted Protein Interactions) is a comprehensive web–based resource for exploring human protein interactions. It integrates data from various interaction databases and stores them in a relational database. It also incorporates a unified scoring scheme to calculate the quality/confidence of the protein interaction results by giving them a star rating ranging from 1 through 5 [165]. As of November 2009, this database contained information for 13,601 proteins and almost 1.3 million PPI (`http://discern.uits.iu.edu:8340/HAPPI/`). BioGRID (`http://thebiogrid.org/`) is another online interaction database that searches over 30,287 publications for 461,097 raw protein and genetic interactions from major model organism species (as of February 2012). The new curated interactions are updated monthly. The interaction data are freely available to download in tab delimited text and PSI–MI XML which

are HUPO's standard to store interactions. IntAct provides freely available database system for containing information(as of February 2012) for 60,993 proteins and over 291,835 binary interactions (`http://www.ebi.ac.uk/intact/`).

5.3 Available algorithms

There are a number of methods that integrate information from different interaction databases to predict gene functions. GeneMANIA[172] [173] is one such tool that integrates multiple functional association networks and predicts gene functions. There are a few approaches that address the issue of intercellular interaction using the available interaction network information. In one study, *Tarca et al.* used signaling pathway impact analysis to gain biological insight from two set of genes [174]. *Kirouac et al.* [175] studied the intercellular and intracellular networks in a stem cell derived, hierarchically organized tissue by analyzing cultured human umbilical cord blood progenitors. They showed that secreted factor-mediated intercellular communication networks regulate blood stem cell fate decisions. However, in none of these studies has a general method to determine possible intermediaries in intercellular signaling been proposed.

5.4 Methodology

A naïve algorithm and the proposed heuristic algorithm are outlined here to solve the above problem. The gene lists are assumed to be from two different tissues, T_1 and T_2 and may be subsets of much larger lists, selected using some criteria such as the level of expression of individual genes. One protein may interact with another either directly or indirectly via a number of other proteins. One direct interaction between two proteins is called a *hop*.

5.4.1 Naïve algorithm

The naïve approach to find all possible interactions between two sets of proteins is to take the protein list T_1 as a seed into the interaction database to find all possible interactions. Using the targets from the previous step as the source nodes, a new set of interactions are found. This step is repeated up

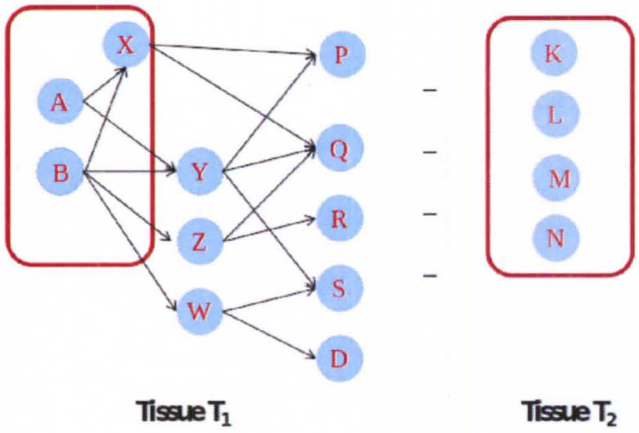

Figure 5.2: Naïve approach to find gene interaction.

to h hops iteratively. Fig. 5.2 illustrates the naïve approach where a gene or protein in tissue T_1 may interact with other genes or proteins, and those in turn interact with others and so on, finally reaching genes in tissue T_2. If the initial gene set is {A, B}, then using the interaction database it can be determined that these two genes in turn interact with four genes {X, Y, Z, W} where A interacts with {X,Y} and B interacts with {X, Y, Z, W} as shown by the edges in Fig. 5.2. Some of these genes may be found to be expressed in the same or different cellular components such as the nucleus, cytoplasm or cellular membrane. Gene X belongs to the same cellular component as {A, B} whereas genes {Y, Z, W} belong to different cellular components. Fig. 5.2 shows the network after two degrees of separation or hops (h) in tissue T_1. In this example, gene A interacts with X and X interacts with P, resulting in two hops to traverse from A to P. The number of hops between a pair of nodes from one tissue to another may vary. To address this concern, the interactions may be checked for a number of hops ranging from 1 through h. The pseudocode for finding all interactions using a naïve algorithm is shown in Algorithm 2. In this example, $intxnDB$ is a PathwayCommons interaction database in simple interaction file (SIF) format where each entry represents an interaction from a source ('from') node to a destination ('to') node with each node being a gene or protein. Although there is no directionality in the actual interaction data, directionality is explicitly added by appending a symmetric property to the data. In this case, if entry A interacts with B (A,B) then B interacts with A (B,A) is added into the set of interactions.

Algorithm 2 Naïve Algorithm for Finding all Interactions.

```
1: procedure AllIntxnNaive(input, h, intxnDB)
2:     n ← length(input);                    ▷ input contains all start nodes (A and B in Fig. 5.2)
3:     hops ← 1
4:     repeat
5:         for j ← 1, n do                   ▷ for each input find all direct interactions
6:             node = input[j]
7:             for k ← 1, lenIntxnDb do      ▷ for each input find all direct interactions
8:                 if node = intxnDB[k, 'from'] then   ▷ Add intxnDB[k, 'to'] as child of node;
9:                 end if
10:            end for
11:        end for                           ▷ input is now all the children attached in the previous step
12:        n = length(input)
13:        hops = hops + 1
14:    until hops ≤ h
15: end procedure
```

Starting with n initial nodes (A and B are two initial nodes in Fig. 5.2), finding all possible interactions requires searching for all interactions of a gene or protein and incrementally building the interaction network. Taking the rat interactome as an example (511,408 interactions and 3,778 nodes as of April 18^{th}, 2011. Average interactions per node = 136), and assuming that the interactions are represented as a tree structure, on average each of the nodes at the root level has 136 children. Each of these children at the first level, on average, has 136 children in the second level, and so on. Finding all the interactions in such a way is an intractable problem as the run time for the algorithm will be $\theta(n*136^h*m)$ where h is the maximum number of hops (levels in tree example) required and m is the total number of interactions in the $intxnDB$. Using a binary search to find interactions in the $intxnDB$ will take $\theta(n*136^h*log_2(m))$ time, where $intxnDB$ is the rat interaction data, and $\theta(n*k^h*log_2(m))$ time in general, when k is the average number of interactions for each node.

5.4.2 Proposed heuristic approach

To address the computational issues with the naïve approach, a heuristic algorithm is proposed to find all possible interactions across tissues in an iterative way. The heuristic uses an exclusion vector (EV) that is updated at each iteration to maintain a list of those nodes that should not be considered in future iterations. The "backtracking approach" removes in each iteration all those nodes already used in the previous iterations. This reduces the complexity of the search space as the removal of a

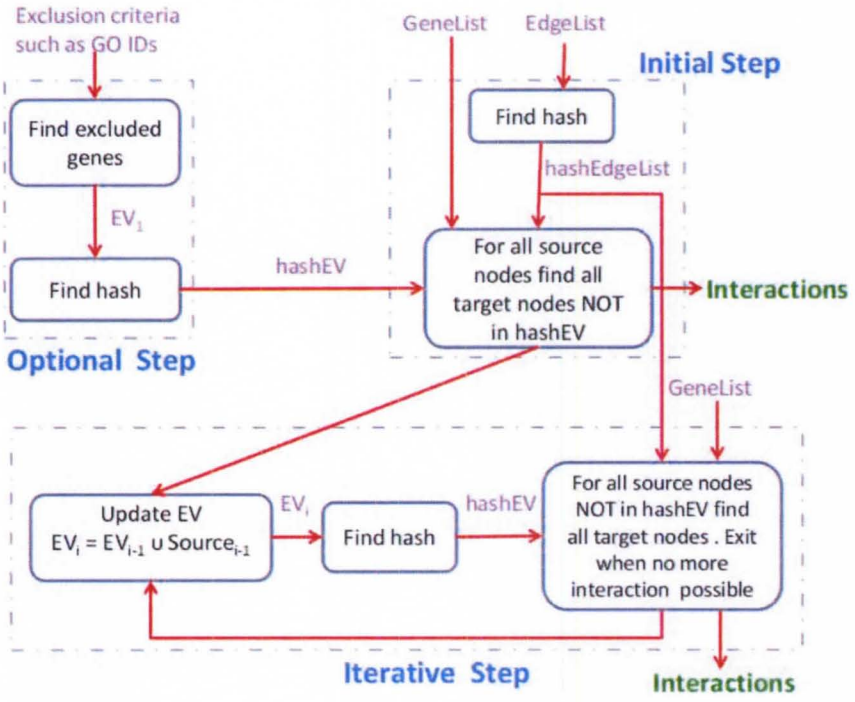

Figure 5.3: Flow diagram for finding participating nodes and interactions.

single node in $136 - ary$ tree removes the subtree rooted at that node, thereby reducing the overall complexity to be directly proportional to the number of participating node p in the final interaction. A *meet-in-the-middle* (MIM) concept is used to limit the number of participating interactions by removing all those nodes that do not lead to MIM nodes. MIM nodes are defined as those nodes found in common at some place between two tissues when the traversal begins from the set of nodes from either tissue. Once the common nodes are known, a trace–back can be used to include only those interactions that lead to MIM nodes, while the rest of the interactions are removed.

The EV can also be used to store a set of nodes that should be excluded from the interaction calculation. The EV can be initialized with an optional set of nodes that are known not to play any role in the interaction. A hash table is used for storing the interactions, thereby reducing the running time to $\theta(n * 136^h * \theta(1))$ in contrast to $\theta(n * 136^h * m)$ in the naïve approach. The complete heuristic algorithm is shown as flow diagram in Fig. 5.3 and the steps are explained as follows.

1. **Location Awareness:** Considering that many proteins are localized to a specific region of the cell, have different molecular functions or are involved in different biological processes

that may restrict the possibility of interacting across tissues, a desirable property is to have some control over which genes are considered. For instance, if it is known that a protein is restricted to the nucleus and is not going to play a role in some form of direct intercellular interaction, it is better to be able to exclude (optionally) all those genes that are in the nucleus and not found anywhere else. One popular source of information for these properties is Gene Ontology (GO), which contains annotated information concerning the cellular localization of proteins. Therefore, it is advantageous to populate the EV based on those genes with/without particular GO annotations as an initial step in the algorithm. However, searching all genes that are exclusively annotated by a subset of GO annotations may take time $\theta(n * m)$, where n is number of genes while m is total number of cell components. In the worst case the complexity will be $\theta(n^2)$. This algorithm stores the annotations as bit vectors to allow quick searching of genes that are annotated with particular GO terms in time $\theta(n)$.

Table 5.1: Occurence matrix using cellular component information for a sample gene set.

		NUCLEUS	CYTOPLASM	MITOCHONDRION	PLASMA MEMBRANE	CELL JUNCTION	SYNAPSE
swissProtID	GeneName	GO: 5634	GO: 5737	GO: 5739	GO: 5886	GO: 30054	GO: 45202
O55007	Park2	1	1	1	0	0	0
Q9ES40	Arl6ip5	0	1	0	0	0	0
P08050	Gja1	0	1	0	1	1	0
B0BNC4	Agxt2l2	0	0	0	0	0	0
O88407	Faim2	0	0	0	1	1	1
B0BNC8	Garnl1	1	1	1	0	0	0
Q8K5C2	Park2	1	1	0	1	1	1
Q9JM59	Kcnip2	0	1	0	1	0	0
O35049	Smpd3	0	0	0	1	0	0

Given a set of gene–GO annotations, an occurrence vector (OV) is generated for each gene with all GO annotations in a given sub-ontology (biological process, molecular function, or cellular component (CC)). An example occurrence matrix (OM) is shown in Table 5.1. The table contains six cell components with GO identifiers 5634, 5737, 5739, 5886, 30054, 45202. The OV for all the genes will be generated once and can be used later. For instance, the OV for the gene park2 (swissProt ID O55007) is {111000} with 0 indicating absence and 1 indicating the presence of the park2 gene product in the corresponding CC. A list of GO identifiers are supplied. Genes that are exclusively annotated with those identifiers or their subsets are selected to serve as candidates for the exclusion vector. For instance, if the supplied GO CC

IDs are {5737, 5886}, then genes are exclusively annotated with 5737 or 5886 or both. The OV corresponding to the query will be $OV_q = \{010100\}$. For each gene i, if $OV_q \otimes OV_i = 0$ then gene i is included in the initial set of exclusion vector EV_1. Here, the symbol $\otimes$ can be defined as:

$$X \otimes Y = \Sigma_{i=1}^{m}(\overline{X_i} \cdot Y_i)$$

where $\overline{X_i}$ represents the bitwise NOT of X_i and $\cdot$ represents the bitwise AND operation. In Table 5.1, only Q9ES40, B0BNC4, Q9JM59, O35049 qualify to be included in EV_1 while the rest of the genes are removed as they are either not annotated with 5737 or 5886 or they are also annotated with additional GO identifiers. Finding genes using the OV will require $\theta(n)$ time as the OV of each gene is compared with the OV_q using a bitwise operation. It should be noted that, in theory, any gene annotation data can be used to generate the OV and subsequently used to populate the EV at the initial step of the algorithm.

2. **Initial Step:** The complete interaction database and two sets of genes (set of nodes from T_1 and T_2) between which interactions are to be determined are given as input. A hash table (*hashEdgeList*) is generated from the edge list. An optional EV_1 from the previous step (EV_1 is empty when location awareness is not taken into consideration) is also converted to a hash (*hashEV*). For each source node a lookup is performed on *hashEdgeList* adding all those edges with targets not in *hashEV*. Once the interactions for all the nodes in the set are found then one hop (pass) is completed (Fig. 5.4a-b).

3. **Iterative Step:** In the iterative step, the EV is first updated. In the i^{th} iteration, EV_i is populated as follows:

$$EV_i = EV_{i-1} \cup src_{i-1}, \quad i \geq 2$$

The *EV* at each step is a union of all the nodes in the *EV* and the source nodes (*src*) in the previous step (Fig 5.3). The EV_i is then converted into a hash table (*hashEV*). Using the interactions from the previous iteration, *hashEV* and *hashEdgeList*, the interactions for each source node not in *hashEV* are found. This step is repeated for the required number of hops or until no more interactions can be added to the system. This will generate all

valid interactions considering the location–aware algorithm combined with the heuristic that iteratively populates the EV.

Algorithm 3 Heuristic algorithm to generate all participating interactions.

```
1: procedure HEURISTIC (geneList, hashEV, edgeList)
2:     hashEdgeList = hash(edgeList[,'from'], edgeList[,'to'])
   ———— Initial step ————
3:     intxn = φ
   /* Store interactions in present hop. */
4:     for i ← 1, length(geneList) do
5:         src = geneList[i]
6:         tgts = hashEdgeList[[src]]
7:         if tgts != NULL then
8:             for j ← 1, length(tgts) do
9:                 tgt = tgts[j]
10:                if hashEV[[tgt]] == NULL then
11:                    Add (src, tgt) to the intxn
12:                end if
13:            end for
14:        end if
15:    end for
   ———— Iterative Step ————
16:    loop = TRUE;
17:    while loop do
18:        src = intxn[,'tgt']
19:        if length(src) ≤ 0 then
20:            loop == FALSE
21:            break
22:        end if
23:        EV = EV ∪ src;
24:        hashEV = invert(hash(1 : length(EV), EV));
25:        for i ← 1, length(src) do
26:            s = src[i]
27:            if hashEV[[s]] != NULL then
28:                tgts = hashEdgeList[[src]]
29:                if tgts != NULL then
30:                    Add (src, tgts) to the intxn
31:                else
32:                    Exit
33:            end if
34:        end if
35:    end for
36: end while
end procedure
```

Pseudocode for the heuristic algorithm is given in Algorithm 3. These interactions are only generated once for a particular set of differentially expressed nodes. Line 2 builds a hash table. Lines 3 to 15 represent the initial step. Lines 16 to 30 iteratively find the interactions.

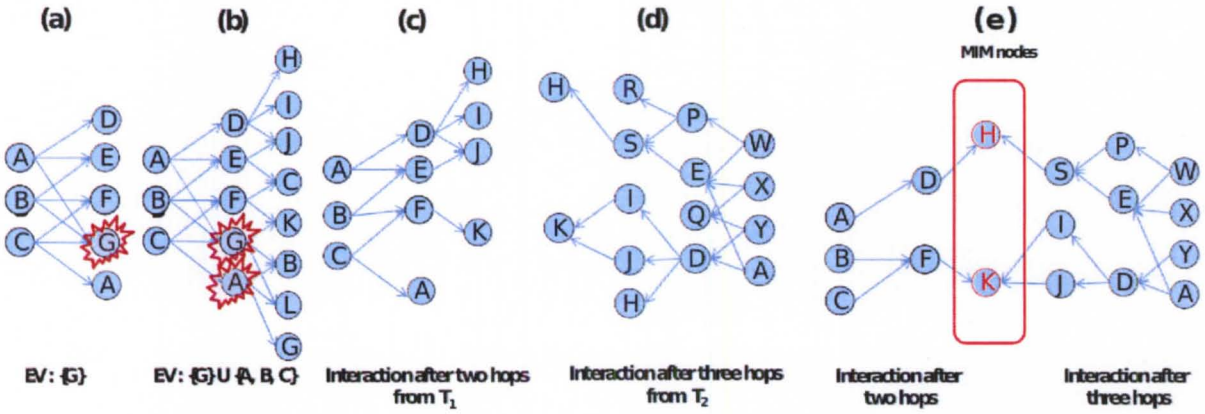

Figure 5.4: Steps in the construction of an interaction network using the heuristic algorithm.

4. Steps 2 and 3 are performed for both gene sets under consideration to find all possible interactions fulfilling the criteria. Fig. 5.4(c) shows two hops of interactions from T_1 while Fig. 5.4(d) shows three hops of interactions from T_2.

5. The MIM nodes are determined, and the edges and nodes that do not lead to MIM nodes are removed. Fig. 5.4(e) shows the interaction network after performing this trace–back. The size of resulting interaction graph is smaller than the one with the full set of interactions. The interacting proteins between the tissues can then be viewed in a graph visualization package such as *Cytoscape* [176] [177].

5.5 Finiteness and completeness of the heuristic approach

The proposed heuristic algorithm is finite, meaning that the algorithm will come to a halt after performing a certain number of steps. The EV is populated at each iteration and its size increases after every iteration. The algorithm converges and ceases when no new nodes can be added into the interaction. At this moment, the EV contains the same set of nodes as those in the network itself. In contrast, the naïve algorithm does not contain any criteria of finiteness and a maximum number of hops (h) must be supplied to the algorithm to force completion. The EV in the i_{th} iteration are updated as below:

$$EV_i = EV_{i-1} \cup src_{i-1}, \qquad i \geq 2$$

Inclusion of all the nodes in src_{i-1} in the equation is guaranteed to include all the back edges and correctly include all the nodes and interactions. The initial step of the algorithm in Fig. 5.3 applies the EV on the target nodes instead of source nodes (iterative step) to keep all nodes that are nuclear and in the *geneList* as the cell interaction may generate from the nucleus and go out to some other cells.

5.6 Results

The following results all use the interaction database for rat from PathwayCommons database. This interactome contains a total of 511,408 interactions with 3,778 nodes as of April 18^{th}, 2011. Each node on an average contains 136 direct interactions with other nodes. Finding interactions using the naïve approach presented in Algorithm 2 quickly becomes intractable. For instance, with a random starting gene set of just 10 initial genes, the first hop for determining interactions takes 35 seconds. The second hop takes 5,820, and the third hop does not complete. Therefore, a modified naïve approach was implemented that uses a hash to speed up retrieval of gene interactions for comparative purposes.

Table 5.2: Comparison of Heuristic and Naïve algorithm.

		Heuristic			naïve using hash		
geneList	# hops	time	#intxn	#nodes	time	#intxn	#nodes
10	7	10.57	388372	1099	91.78	2714869	1107
50	7	11.26	388384	1108	83.54	2343199	1153
100	6	11.66	388420	1131	70.41	1962689	1220
200	7	11.89	388516	1196	90.92	2376875	1361
300	6	11.76	388478	1180	75.39	2016790	1449
400	5	11.76	388554	1216	114.61	2821283	1583
500	5	11.36	388565	1224	63.93	1674262	1692

The speed-up of the heuristic algorithm versus the modified naïve approach with hashing was computed by taking random gene lists (ranging from 10 to 500 in number of genes) extracted from the PathwayCommons rat interactome dataset for both the "from" and "to" gene lists which conceptually represent different tissue types. The generated gene list size is given in the *geneList* field of Table 5.2. An additional input for the number of maximum hops must be supplied for

the naïve algorithm, as it will otherwise continue adding interactions *ad infinitum*. For each of the gene lists, the maximum number of hops used was the number of hops taken by the heuristic algorithm before completion to provide a fair comparison of the approaches. Since the gene lists are generated randomly, the number of required hops fluctuates between five and seven as shown in the Table 5.2. In these examples the EV was used only to remove nodes that had been previously encountered (i.e. no localization information was included). As can be seen in Table 5.2, the number of nodes (i.e. genes) and the number of interactions both remain relatively stable with the heuristic approach, while the naïve algorithm has greater fluctuations. However, the number of interactions increases substantially in the naïve approach, ranging from four to seven times as high as the number of interactions found using the EV heuristic. The actual computational time for generating the respective interaction networks is given in the third column. From these results, the heuristic approach is anywhere between five to nine times faster than the modified naïve approach, which could be reduced further when localization is incorporated in the EV.

To further illustrate the applicability of the heuristic algorithm for intercellular signaling, we generated gene lists from different tissues using a publicly available dataset from Gene Expression Omnibus (GEO) [93] (GEO accession GDS1864, http://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS1864) containing 62 total samples studying the effects of two antiepileptic drugs (levetiracetam, phenytoin) on the expression of genes in three brain tissues: brainstem, frontal cortex, and hippocampus. As the drugs are administered to whole animals, it is possible that some of the changes in gene expression are due to intercellular signaling between the tissues. Differentially expressed genes are found using *empirical Bayes* statistics [178] from the Bioconductor [179] *limma* [55] package with a p-value < 0.05. The initial gene lists contained 311 and 324 differentially expressed genes for the frontal cortex and the hippocampus tissues respectively. Up– and down– regulated gene lists for each tissues logFC > 0.5 for up–regulated, logFC < -0.5 for down–regulated genes) were generated for the comparison of exposure to phenytoin with controls. The interactions between up–regulated genes in cortex and up–regulated genes in hippocampus are shown in Fig. 5.5. Four hops from the frontal cortex and three hops from the hippocampus (a total of six intermediate nodes between cortex and

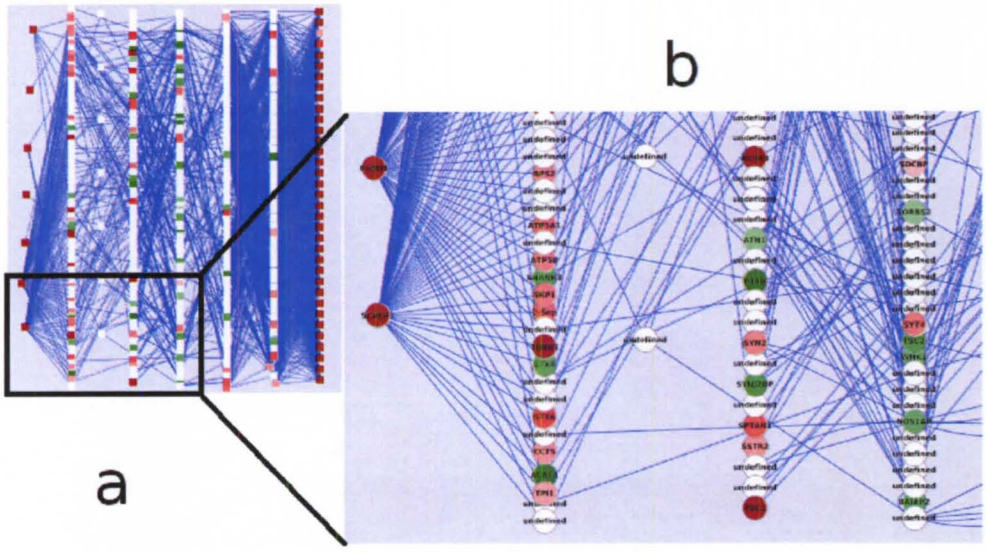

Figure 5.5: (a). Output from *Cytoscape* showing interactions between the frontal cortex (left) and the hippocampus (right). (b). Detail of the inset in (a).

hippocampus) are considered here. Nodes (genes) up–regulated in the respective tissue are colored red, while those that are down–regulated are green. Nodes that are white represent genes that are either not significantly changed, or are not present on the array. The resulting interaction network contains a total of 387 nodes (genes) and 2170 interactions. This example demonstrates how an interaction network can be built, and subsequently visualized. Gene location information was not used in this example. Inclusion of location information and further filtering by an expression cut-off can significantly reduce the interaction network even further.

5.7 Conclusion

A heuristic algorithm is developed for detecting and predicting intercellular interactions. Considering the large number of interactions this algorithm may serve as a time efficient algorithm to view interactions between cells. The use of the EV allows location awareness in the interaction in an efficient manner. The MIM approach further limits the size of the interaction network without losing any information. The success of the method depends upon the information available in the respective databases. The more accurate the database, the more reliable the output network will

be. In this work the edge attributes are not considered, however their inclusion may increase the confidence in the generated interaction network.

CHAPTER 6

SUMMARY AND FUTURE WORK

High–throughput techniques in molecular biology are generating high volumes of data waiting to be understood. A number of databases and software tools have been created to deal with these data. Oftentimes these databases and their methods for annotating biological entities are independent, heterogeneous and redundant, yet at the same time contain important information. When combined, these sources of information can provide a better understanding of a biological system as a whole. Integration or comparison of these databases is difficult, time–consuming and sometimes impossible because of the absence of a common platform to compare them directly.

To mitigate issues with conversion among annotations, we developed AbsIDconvert, an absolute ID conversion tool. This tool is absolute in that it converts annotations to a common source based on underlying cytogenetic locations. These are represented as intervals with definite start and end locations, exonic boundaries and lengths. AbsIDconvert uses an efficient interval–tree approach to store the coordinate–level information and is effective in integrating and comparing heterogeneous databases. To our knowledge, AbsIDconvert is the only known gene ID conversion tool based on genomic coordinates. AbsIDconvert provides ease of flexibility with respect to any additions, deletions or updates of the underlying objects, requiring only adding of intervals, removing intervals, or modifying the intervals themselves, respectively.

AbsIDconvert allows flexibility in specifying the overlapping parameters while performing ID conversion. It can discover partially overlapped IDs / intervals, or those which are within a specified distance nearby. It also allows discovery of overlapping IDs for a given set of sequences and intervals. With the incorporation of a sequence mapping algorithm, AbsIDconvert allows the determination of genomic intervals for any supplied sequence, making it possible to efficiently find and convert

between IDs from any platform and organism, as demonstrated by the case studies in chapter 4. AbsIDconvert is also helpful in meta analyses studies such as for performing cross–species and / or cross–platform studies (case study 3). All these functionalities of AbsIDconvert give it a clear advantage over other available tools.

AbsIDconvert is currently available at `http://bioinformatics.louisville.edu/abid/` with support to analyze data for 53 organisms, containing a total of over 50 million identifiers. Full support for additional 1497 bacterial strains are also available.

Since annotations are dynamic, AbsIDconvert also require regular updates; however it is stable compared to annotation based tools. An update is required when a genome is updated or the DNA sequence for an entity changes which is not so frequent. Future work includes providing command line access, modifying the interface to utilize a workflow management tool for genomic data such as GALAXY, development of fully automated updates, and support for other genomes including plant, microbial and viral genomes.

Current high–throughput gene expression analyses treat data as if they are obtained from a single or homogeneous cell population and account only for *intracellular* interactions. However, *intercellular* interactions are equally important and are generally ignored in these analyses. To account for the interplay between different cells or tissues, we developed a heuristic algorithm for detecting and predicting intercellular interactions between two populations of interest using publicly available interaction datasets. Our tractable heuristic algorithm incorporates location awareness at each iteration using GO ontological cellular component information. An exclusion vector (EV) is used that efficiently keeps only those interactions that are relevant by restricting the search space based on defined properties of the genes. An MIM (meet–in–the–middle) criteria is also applied to further limits the size of the interaction network without losing any information. This method has been applied to find interactions between frontal cortex and hippocampus tissues as well as skin and DRG data from Dr. Jeff Petruska's lab. This method can be readily applied to separate tissue samples that interact, such as neuronal cell bodies and their target tissues, or specific cell-types separated from their native tissue.

The success of the method depends upon the accuracy of interaction information available in public databases. The current method considers all interactions to be equally probable. However, in an actual biological system, this assumption is limiting. A future improvement to this work would be to include weight or probability information on each interaction which would lead to more accurate detection of protein interactions. Using this *a priori* information, a Bayesian algorithm can be applied to find the posterior. Additionally, movement of signaling information from one tissue to another may be systematically determined by considering a hierarchical system based on localization such as cellular component. For instance, a signal that moves from cell A to another cell B, will always have to pass from cytoplasm of A, plasma membrane of A, extracellular matrix, plasma membrane of B and cytoplasm of B. This definitive path may be used instead of MIM nodes to get more accurate results.

The maintenance of a steady state in complex organisms requires individual cells to perform activities in a coordinated manner. Ignoring communication of these components while performing gene expression analysis and assuming that expression is isolated, does not give a complete picture to biological system as a whole. The systems–based approach proposed in this dissertation overcome these limitations by taking into account the complete coordinated system. This approach is further enhanced by AbsIDconvert that considers all annotations to be sequence–based. These methods will further advance our knowledge of biological systems at a molecular level by looking at the gene expression data in a more plausible manner.

REFERENCES

[1] Information transfer along sensory neuron. http://www.yachigusaryu.com/blog/2006/08/top-ten-principles-of-yachigusa-ryu_18.html.

[2] Michael Y Galperin and Xosé M Fernández-Suárez. The 2012 nucleic acids research database issue and the online molecular biology database collection. *Nucleic Acids Res*, 40(1):D1–8, January 2012.

[3] Alexander Zien. A primer on molecular biology. 2004.

[4] Biology introduction at european bioinformatics institute. http://www.ebi.ac.uk/microarray/biology_intro.html.

[5] Tara Rodden Robinson. *Genetics For Dummies.* Wiley Publishing, Inc, first edition, 2005.

[6] Genetics and molecular biology primer. http://members.cox.net/amgough/Fanconi-genetics-genetics-primer.htm.

[7] Chemical structure of a dna molecule. http://en.wikipedia.org/wiki/File:DNA_chemical_structure.svg.

[8] James D. Wastson. *DNA: The Secret of Life.* Alfred A. Knopf, first edition, 2006.

[9] James D. Watson and Francis H. C. Crick. Molecular structure of nucleic acid. *Nature*, (4356):737, april, 1953.

[10] Dna replication. http://www.replicationfork.com/.

[11] Rene Fester Kratz. *Molecular and Cell Biology For Dummies.* Wiley Publishing, Inc, first edition, 2009.

[12] Levels of protein structure. http://en.wikipedia.org/wiki/File:Main_protein_structure_levels_en.svg.

[13] Francis H. C. Crick. Ideas on protein synthesis. In *Symposia of the Society for Experimental Biology XII*, October 1956.

[14] Francis H. C. Crick. Central dogma of molecular biology. *Nature*, 227:561–563, August, 1970.

[15] The International Human Genome Mapping Consortium. A physical map of the human genome. *Nature*, 409:934–941, February, 2001.

[16] Alternative splicing at wikimedia commons. http://commons.wikimedia.org/wiki/File:AlternativeSplicing.png.

[17] Jonathan Pevsner. *Bioinformatics and Functional Genomics.* Wiley–Liss, first edition, 2003.

[18] Harvey Lodish, Arnold Berk, S. Lawrence Zipursky, Paul Matsudaira, and David Baltimore. Molecular cell biology (4th edition). *Biochemistry and Molecular Biology Education*, pages 126–128, 2001.

[19] Suzanne Clancy and William Brown. *Translation: DNA to mRNA to protein*, chapter 1. Nature Education, 1 edition, 2008.

[20] Misook Ha, Mingxiong Pang, Vikram Agarwal, and Z Jeffrey Chen. Interspecies regulation of micrornas and their targets. *Biochim Biophys Acta*, 1779(11):735–42, November 2008.

[21] Joseph Sambrook and David W Russell. Fragmentation of dna by sonication. *CSH Protoc*, 2006(4), 2006.

[22] Eric C. Rouchka. Bioinformatics lecture slides. Fall, 2009.

[23] A.M. Maxam and W. Gilbert. A new method for sequencing dna. *Proceedings of the National Academy of Sciences*, 74(2):560, 1977.

[24] F. Sanger and A.R. Coulson. A rapid method for determining sequences in dna by primed synthesis with dna polymerase. *Journal of molecular biology*, 94(3):441–448, 1975.

[25] F Sanger, S Nicklen, and A R Coulson. Dna sequencing with chain-terminating inhibitors. *Proc Natl Acad Sci U S A*, 74(12):5463–7, December 1977.

[26] L M Smith, J Z Sanders, R J Kaiser, P Hughes, C Dodd, C R Connell, C Heiner, S B Kent, and L E Hood. Fluorescence detection in automated dna sequence analysis. *Nature*, 321(6071):674–9, June 1986.

[27] Michael L Metzker. Emerging technologies in dna sequencing. *Genome Res*, 15(12):1767–1776, Dec 2005.

[28] Elaine R Mardis. Next-generation dna sequencing methods. *Annu Rev Genomics Hum Genet*, 9:387–402, 2008.

[29] Overview of solid^{TM}sequencing chemistry. Technical report.

[30] John Eid, Adrian Fehr, Jeremy Gray, Khai Luong, John Lyle, Geoff Otto, Paul Peluso, David Rank, Primo Baybayan, Brad Bettman, Arkadiusz Bibillo, Keith Bjornson, Bidhan Chaudhuri, Frederick Christians, Ronald Cicero, Sonya Clark, Ravindra Dalal, Alex Dewinter, John Dixon, Mathieu Foquet, Alfred Gaertner, Paul Hardenbol, Cheryl Heiner, Kevin Hester, David Holden, Gregory Kearns, Xiangxu Kong, Ronald Kuse, Yves Lacroix, Steven Lin, Paul Lundquist, Congcong Ma, Patrick Marks, Mark Maxham, Devon Murphy, Insil Park, Thang Pham, Michael Phillips, Joy Roy, Robert Sebra, Gene Shen, Jon Sorenson, Austin Tomaney, Kevin Travers, Mark Trulson, John Vieceli, Jeffrey Wegener, Dawn Wu, Alicia Yang, Denis Zaccarin, Peter Zhao, Frank Zhong, Jonas Korlach, and Stephen Turner. Real-time dna sequencing from single polymerase molecules. *Science*, 323(5910):133–138, Jan 2009.

[31] Michael L Metzker. Sequencing in real time. *Nat Biotechnol*, 27(2):150–151, Feb 2009.

[32] Michael L Metzker. Sequencing technologies - the next generation. *Nat Rev Genet*, 11(1):31–46, Jan 2010.

[33] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J Mol Biol*, 215(3):403–410, Oct 1990.

[34] W. James Kent. Blat–the blast-like alignment tool. *Genome Res*, 12(4):656–664, Apr 2002.

[35] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L Salzberg. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biol*, 10(3):R25, 2009.

[36] Ben Langmead. Aligning short sequencing reads with bowtie. *Curr Protoc Bioinformatics*, Chapter 11:Unit 11.7, December 2010.

[37] Heng Li, Jue Ruan, and Richard Durbin. Mapping short dna sequencing reads and calling variants using mapping quality scores. *Genome Res*, 18(11):1851–8, November 2008.

[38] AJ Cox. Eland: efficient local alignment of nucleotide data. *unpublished, http://bioit. dbi. udel. edu/howto/eland*, 2006.

[39] Ruiqiang Li, Yingrui Li, Karsten Kristiansen, and Jun Wang. Soap: short oligonucleotide alignment program. *Bioinformatics*, 24(5):713–714, Mar 2008.

[40] Ruiqiang Li, Chang Yu, Yingrui Li, Tak-Wah Lam, Siu-Ming Yiu, Karsten Kristiansen, and Jun Wang. Soap2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25(15):1966–1967, Aug 2009.

[41] Daniel R. Zerbino and Ewan Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome Research*, 18:821–829, 2008.

[42] Sante Gnerre, Iain Maccallum, Dariusz Przybylski, Filipe J Ribeiro, Joshua N Burton, Bruce J Walker, Ted Sharpe, Giles Hall, Terrance P Shea, Sean Sykes, Aaron M Berlin, Daniel Aird, Maura Costello, Riza Daza, Louise Williams, Robert Nicol, Andreas Gnirke, Chad Nusbaum, Eric S Lander, and David B Jaffe. High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc Natl Acad Sci U S A*, 108(4):1513–8, January 2011.

[43] Douglas W Bryant, Weng-Keen Wong, and Todd C Mockler. Qsra: a quality-value guided de novo short read assembler. *BMC Bioinformatics*, 10:69, 2009.

[44] William R Jeck, Josephine A Reinhardt, David A Baltrus, Matthew T Hickenbotham, Vincent Magrini, Elaine R Mardis, Jeffery L Dangl, and Corbin D Jones. Extending assembly of short dna sequences to handle error. *Bioinformatics*, 23(21):2942–4, November 2007.

[45] Kary Mullis *et al.* Specific enzymatic amplification of DNA in vitro: The polymerase chain reaction. *Cold Spring Harbor Symposia on quantitative biology*, 51:263–273, 1986.

[46] The Celera Genomics Sequencing Team. The sequence of the human genome. *Science*, pages 1304–1351, February, 2001.

[47] Shivashankar H Nagaraj, Robin B Gasser, and Shoba Ranganathan. A hitchhiker's guide to expressed sequence tag (est) analysis. *Brief Bioinform*, 8(1):6–21, Jan 2007.

[48] Dov Stekel. *Microarray Bioinformatics*. Cambridge University Press, first edition, 2003.

[49] Microarray analysis steps. `http://www.biotools.eu/imagenes/FOTO1-ESQUEMA.gif`.

[50] John Quackenbush. Microarray data normalization and transformation. *Nat Genet*, 32 Suppl:496–501, Dec 2002.

[51] V. G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proc Natl Acad Sci U S A*, 98(9):5116–5121, Apr 2001.

[52] Ramón Díaz-Uriarte and Sara Alvarez de Andrés. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7:3, 2006.

[53] Jiaqin Liu, Jianniao Tian, Ying Li, Xiaojun Yao, Zhide Hu, and Xingguo Chen. Binding of the bioactive component daphnetin to human serum albumin demonstrated using tryptophan fluorescence quenching. *Macromol Biosci*, 4(5):520–525, May 2004.

[54] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of Royal Statistical Society*, 57(1):289–300, 1995.

[55] Gordon K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions using R and Bioconductor*, pages 397–420. Springer, New York, 2005.

[56] K. Pollard, S. Dudoit, and M. Laan. Multiple testing procedures: the multtest package and applications to genomics. In Robert Gentleman, Vincent J. Carey, Wolfgang Huber, Rafael A. Irizarry, and Sandrine Dudoit, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, Statistics for Biology and Health, pages 249–271. Springer New York, 2005. 10.1007/0-387-29362-0_15.

[57] Stefanie Scheid and Rainer Spang. twilight; a bioconductor package for estimating the local false discovery rate. *Bioinformatics*, 21(12):2921–2, June 2005.

[58] Nema Dean and Adrian E Raftery. Normal uniform mixture differential gene expression detection for cdna microarrays. *BMC Bioinformatics*, 6:173, 2005.

[59] Natalia Becker, Wiebke Werft, Grischa Toedt, Peter Lichter, and Axel Benner. penalizedsvm: a r-package for feature selection svm classification. *Bioinformatics*, 25(13):1711–2, July 2009.

[60] Sorin Draghici, Purvesh Khatri, Pratik Bhavsar, Abhik Shah, Stephen A Krawetz, and Michael A Tainsky. Onto-tools, the toolkit of the modern biologist: Onto-express, onto-compare, onto-design and onto-translate. *Nucleic Acids Res*, 31(13):3775–81, July 2003.

[61] A. Brazma, P. Hingamp, J. Quackenbush, G. Sherlock, P. Spellman, C. Stoeckert, J. Aach, W. Ansorge, C. A. Ball, H. C. Causton, T. Gaasterland, P. Glenisson, F. C. Holstege, I. F. Kim, V. Markowitz, J. C. Matese, H. Parkinson, A. Robinson, U. Sarkans, S. Schulze-Kremer, J. Stewart, R. Taylor, J. Vilo, and M. Vingron. Minimum information about a microarray experiment (miame)-toward standards for microarray data. *Nat Genet*, 29(4):365–371, Dec 2001.

[62] M Madan Babu. Introduction to microarray data analysis. In *Computational Genomics: Theory and Application*. Horizon Press, 2004.

[63] Microarrays: Chipping away at the mysteries of science and medicine. `http://www.ncbi.nlm.nih.gov/About/primer/microarrays.html`.

[64] L Stein. Genome annotation: from sequence to biology. *Nat Rev Genet*, 2(7):493–503, July 2001.

[65] C B Burge and S Karlin. Finding the genes in genomic dna. *Curr Opin Struct Biol*, 8(3):346–54, June 1998.

[66] G Parra, E Blanco, and R Guigo. Geneid in drosophila. *Genome Res*, 10(4):511–5, April 2000.

[67] A L Delcher, D Harmon, S Kasif, O White, and S L Salzberg. Improved microbial gene identification with glimmer. *Nucleic Acids Res*, 27(23):4636–41, December 1999.

[68] Mark Borodovsky and Alex Lomsadze. Gene identification in prokaryotic genomes, phages, metagenomes, and est sequences with genemarks suite. *Curr Protoc Bioinformatics*, Chapter 4:Unit 4.5.1–17, September 2011.

[69] Gunnar Ratsch, Saren Sonnenburg, Jagan Srinivasan, Hanh Witte, Klaus-R Muller, Ralf J Sommer, and Bernhard Scholkopf. Improving the caenorhabditis elegans genome annotation using machine learning. *PLoS Comput Biol*, 3(2):e20, February 2007.

[70] Samuel S Gross, Chuong B Do, Marina Sirota, and Serafim Batzoglou. Contrast: a discriminative, phylogeny-free approach to multiple informant de novo gene prediction. *Genome Biol*, 8(12):R269, 2007.

[71] Gabriele Schweikert, Jonas Behr, Alexander Zien, Georg Zeller, Cheng Soon Ong, SÃren Sonnenburg, and Gunnar RÃtsch. mgene.web: a web service for accurate computational gene finding. *Nucleic Acids Res*, 37(Web Server issue):W312–6, July 2009.

[72] T M Lowe and S R Eddy. trnascan-se: a program for improved detection of transfer rna genes in genomic sequence. *Nucleic Acids Res*, 25(5):955–64, March 1997.

[73] Karin Lagesen, Peter Hallin, Einar Andreas Rodland, Hans-Henrik Staerfeldt, Torbjarn Rognes, and David W Ussery. Rnammer: consistent and rapid annotation of ribosomal rna genes. *Nucleic Acids Res*, 35(9):3100–8, 2007.

[74] Jana Hertel and Peter F Stadler. Hairpins in a haystack: recognizing microrna precursors in comparative genomics data. *Bioinformatics*, 22(14):e197–202, July 2006.

[75] Shay Artzi, Adam Kiezun, and Noam Shomron. mirnaminer: a tool for homologous microrna gene search. *BMC Bioinformatics*, 9:39, 2008.

[76] E Wingender, X Chen, R Hehl, H Karas, I Liebich, V Matys, T Meinhardt, M Pruess, I Reuter, and F Schacherer. Transfac: an integrated system for gene expression regulation. *Nucleic Acids Res*, 28(1):316–9, January 2000.

[77] Christian J A Sigrist, Lorenzo Cerutti, Nicolas Hulo, Alexandre Gattiker, Laurent Falquet, Marco Pagni, Amos Bairoch, and Philipp Bucher. Prosite: a documented database using patterns and profiles as motif descriptors. *Brief Bioinform*, 3(3):265–74, September 2002.

[78] UniProt Consortium. Ongoing and future developments at the universal protein resource. *Nucleic Acids Res*, 39(Database issue):D214–9, January 2011.

[79] Marco Punta, Penny C Coggill, Ruth Y Eberhardt, Jaina Mistry, John Tate, Chris Boursnell, Ningze Pang, Kristoffer Forslund, Goran Ceric, Jody Clements, Andreas Heger, Liisa Holm, Erik L L Sonnhammer, Sean R Eddy, Alex Bateman, and Robert D Finn. The pfam protein families database. *Nucleic Acids Res*, 40(Database issue):D290–301, January 2012.

[80] Cathy H Wu, Hongzhan Huang, Leslie Arminski, Jorge Castro-Alvear, Yongxing Chen, Zhang-Zhi Hu, Robert S Ledley, Kali C Lewis, Hans-Werner Mewes, Bruce C Orcutt, Baris E Suzek, Akira Tsugita, C R Vinayaka, Lai-Su L Yeh, Jian Zhang, and Winona C Barker. The protein information resource: an integrated public resource of functional annotation of proteins. *Nucleic Acids Res*, 30(1):35–7, January 2002.

[81] Helen Berman, Kim Henrick, and Haruki Nakamura. Announcing the worldwide protein data bank. *Nat Struct Biol*, 10(12):980, December 2003.

[82] The Gene Ontology Consortium. The gene ontology: enhancements for 2011. *Nucleic Acids Res*, 40(D1):D559–D564, January 2012.

[83] Paul Flicek, M Ridwan Amode, Daniel Barrell, Kathryn Beal, Simon Brent, Denise Carvalho-Silva, Peter Clapham, Guy Coates, Susan Fairley, Stephen Fitzgerald, Laurent Gil, Leo Gordon, Maurice Hendrix, Thibaut Hourlier, Nathan Johnson, Andreas K Kähäri, Damian Keefe, Stephen Keenan, Rhoda Kinsella, Monika Komorowska, Gautier Koscielny, Eugene Kulesha, Pontus Larsson, Ian Longden, William McLaren, Matthieu Muffato, Bert Overduin, Miguel Pignatelli, Bethan Pritchard, Harpreet Singh Riat, Graham R S Ritchie, Magali Ruffier, Michael Schuster, Daniel Sobral, Y Amy Tang, Kieron Taylor, Stephen Trevanion, Jana Vandrovcova, Simon White, Mark Wilson, Steven P Wilder, Bronwen L Aken, Ewan Birney, Fiona Cunningham, Ian Dunham, Richard Durbin, XosA M Fernandez-Suarez, Jennifer Harrow, Javier Herrero, Tim J P Hubbard, Anne Parker, Glenn Proctor, Giulietta Spudich, Jan Vogel, Andy Yates, Amonida Zadissa, and Stephen M J Searle. Ensembl 2012. *Nucleic Acids Res*, 40(Database issue):D84–90, January 2012.

[84] Ruth L Seal, Susan M Gordon, Michael J Lush, Mathew W Wright, and Elspeth A Bruford. genenames.org: the hgnc resources in 2011. *Nucleic Acids Res*, 39(Database issue):D514–9, January 2011.

[85] Dennis A Benson, Ilene Karsch-Mizrachi, Karen Clark, David J Lipman, James Ostell, and Eric W Sayers. Genbank. *Nucleic Acids Res*, 40(Database issue):D48–53, January 2012.

[86] D R Maglott, K S Katz, H Sicotte, and K D Pruitt. Ncbi's locuslink and refseq. *Nucleic Acids Res*, 28(1):126–8, January 2000.

[87] Michele Magrane and Uniprot Consortium. Uniprot knowledgebase: a hub of integrated protein data. *Database (Oxford)*, 2011:bar009, 2011.

[88] Donna Maglott, Jim Ostell, Kim D Pruitt, and Tatiana Tatusova. Entrez gene: gene-centered information at ncbi. *Nucleic Acids Res*, 39(Database issue):D52–D57, Jan 2011.

[89] Stephen Rudd. Expressed seuence tags: Alternative or complement to whole genome sequences. *Trends in Plant Science*, 8(7):321–329, July, 2003.

[90] Ests: Gene discovery made easier. `http://www.ncbi.nlm.nih.gov/About/primer/est.html`.

[91] Fabien Campagne and Lucy Skrabanek. Mining expressed sequence tags identifies cancer markers of clinical interest. *BMC Bioinformatics*, 7:481, 2006.

[92] Donna Karolchik, Angie S Hinrichs, and W. James Kent. The ucsc genome browser. *Curr Protoc Bioinformatics*, Chapter 1:Unit1.4, Dec 2009.

[93] Tanya Barrett, Dennis B Troup, Stephen E Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F Kim, Maxim Tomashevsky, Kimberly A Marshall, Katherine H Phillippy, Patti M Sherman, Rolf N Muertter, Michelle Holko, Oluwabukunmi Ayanbule, Andrey Yefanov, and Alexandra Soboleva. Ncbi geo: archive for functional genomics data sets–10 years on. *Nucleic Acids Res*, 39(Database issue):D1005–D1010, Jan 2011.

[94] Ron Edgar, Michael Domrachev, and Alex E Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic Acids Res*, 30(1):207–10, jan 2002.

[95] Affymetrix genechips. `http://www.u.arizona.edu/~gwatts/azcc/Affymetrix.jpg`.

[96] Affymetrix technical notes. `http://www.affymetrix.com/support/technical/index.affx`.

[97] Guoying Liu, Ann E Loraine, Ron Shigeta, Melissa Cline, Jill Cheng, Venu Valmeekam, Shaw Sun, David Kulp, and Michael A Siani-Rose. Netaffx: Affymetrix probesets and annotations. *Nucleic Acids Res*, 31(1):82–6, January 2003.

[98] Pauline A Fujita, Brooke Rhead, Ann S Zweig, Angie S Hinrichs, Donna Karolchik, Melissa S Cline, Mary Goldman, Galt P Barber, Hiram Clawson, Antonio Coelho, Mark Diekhans, Timothy R Dreszer, Belinda M Giardine, Rachel A Harte, Jennifer Hillman-Jackson, Fan Hsu, Vanessa Kirkup, Robert M Kuhn, Katrina Learned, Chin H Li, Laurence R Meyer, Andy Pohl, Brian J Raney, Kate R Rosenbloom, Kayla E Smith, David Haussler, and W. James Kent. The ucsc genome browser database: update 2011. *Nucleic Acids Res*, 39(Database issue):D876–D882, Jan 2011.

[99] J.F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[100] Thomas H. Cormen. *Introduction to algorithms.* MIT Press, Cambridge, Mass., 3rd edition, 2009.

[101] Franco P. Preparata and Michael Ian Shamos. *Computational geometry : an introduction.* Texts and monographs in computer science. Springer, New York, 1985. 85008049 Franco P. Preparata, Michael Ian Shamos. ill. ; 25 cm. Includes indexes. Bibliography: p. [374]-384.

[102] L. Arge and J.S. Vitter. Optimal dynamic interval management in external memory. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 560–569. IEEE, 1996.

[103] Lars Arge and Jeffrey Vitter. Optimal external memory interval management. *SIAM J. Comput.*, 32:1488–1508, June 2003.

[104] M. S. Boguski, T. M. Lowe, and C. M. Tolstoshev. dbest–database for "expressed sequence tags". *Nat Genet*, 4(4):332–333, Aug 1993.

[105] P. Aboyoun, H. Pages, and M. Lawrence. Genomicranges: Representation and manipulation of genomic intervals. *R package version*, 1(6), 2010.

[106] Ada Hamosh, Alan F Scott, Joanna S Amberger, Carol A Bocchini, and Victor A McKusick. Online mendelian inheritance in man (omim), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Res*, 33(Database issue):D514–7, January 2005.

[107] T S Keshava Prasad, Kumaran Kandasamy, and Akhilesh Pandey. Human protein reference database and human proteinpedia as discovery tools for systems biology. *Methods Mol Biol*, 577:67–79, 2009.

[108] Suraj Peri, J Daniel Navarro, Ramars Amanchy, Troels Z Kristiansen, Chandra Kiran Jonnalagadda, Vineeth Surendranath, Vidya Niranjan, Babylakshmi Muthusamy, T K B Gandhi, Mads Gronborg, Nieves Ibarrola, Nandan Deshpande, K Shanker, H N Shivashankar, B P Rashmi, M A Ramya, Zhixing Zhao, K N Chandrika, N Padma, H C Harsha, A J Yatish, M P Kavitha, Minal Menezes, Dipanwita Roy Choudhury, Shubha Suresh, Neelanjana Ghosh, R Saravana, Sreenath Chandran, Subhalakshmi Krishna, Mary Joy, Sanjeev K Anand, V Madavan, Ansamma Joseph, Guang W Wong, William P Schiemann, Stefan N Constantinescu, Lily Huang, Roya Khosravi-Far, Hanno Steen, Muneesh Tewari, Saghi Ghaffari, Gerard C Blobe, Chi V Dang, Joe G N Garcia, Jonathan Pevsner, Ole N Jensen, Peter Roepstorff, Krishna S Deshpande, Arul M Chinnaiyan, Ada Hamosh, Aravinda Chakravarti, and Akhilesh Pandey. Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Res*, 13(10):2363–71, October 2003.

[109] L G Wilming, J G R Gilbert, K Howe, S Trevanion, T Hubbard, and J L Harrow. The vertebrate genome annotation (vega) database. *Nucleic Acids Res*, 36(Database issue):D753–60, January 2008.

[110] Camille Laibe and Nicolas Le Novère. Miriam resources: tools to generate and resolve robust cross-references in systems biology. *BMC Syst Biol*, 1:58, 2007.

[111] M Kanehisa and S Goto. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, 28(1):27–30, January 2000.

[112] Minoru Kanehisa, Susumu Goto, Yoko Sato, Miho Furumichi, and Mao Tanabe. Kegg for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Res*, 40(Database issue):D109–14, January 2012.

[113] Laurent Gautier, Morten Møller, Lennart Friis-Hansen, and Steen Knudsen. Alternative mapping of probes to genes for affymetrix chips. *BMC Bioinformatics*, 5:111, Aug 2004.

[114] Hongfang Liu, Barry R Zeeberg, Gang Qu, A. Gunes Koru, Alessandro Ferrucci, Ari Kahn, Michael C Ryan, Antej Nuhanovic, Peter J Munson, William C Reinhold, David W Kane, and John N Weinstein. Affyprobeminer: a web resource for computing or retrieving accurately redefined affymetrix probe sets. *Bioinformatics*, 23(18):2385–2390, Sep 2007.

[115] Jeremy Harbig, Robert Sprinkle, and Steven A Enkemann. A sequence-based identification of the genes detected by probesets on the affymetrix u133 plus 2.0 array. *Nucleic Acids Res*, 33(3):e31, 2005.

[116] Glynn Dennis, Brad T Sherman, Douglas A Hosack, Jun Yang, Wei Gao, H Clifford Lane, and Richard A Lempicki. David: Database for annotation, visualization, and integrated discovery. *Genome Biol*, 4(5):P3, 2003.

[117] Da Wei Huang, Brad T Sherman, Xin Zheng, Jun Yang, Tomozumi Imamichi, Robert Stephens, and Richard A Lempicki. Extracting biological meaning from large gene lists with david. *Curr Protoc Bioinformatics*, Chapter 13:Unit 13.11, Sep 2009.

[118] Brad T Sherman, Da Wei Huang, Qina Tan, Yongjian Guo, Stephan Bour, David Liu, Robert Stephens, Michael W Baseler, H. Clifford Lane, and Richard A Lempicki. David knowledgebase: a gene-centered database integrating heterogeneous gene annotation resources to facilitate high-throughput gene functional analysis. *BMC Bioinformatics*, 8:426, 2007.

[119] Da Wei Huang, Brad T Sherman, Robert Stephens, Michael W Baseler, H Clifford Lane, and Richard A Lempicki. David gene id conversion tool. *Bioinformation*, 2(10):428–30, 2008.

[120] Douglas A Hosack, Glynn Dennis, Brad T Sherman, H. Clifford Lane, and Richard A Lempicki. Identifying biological themes within lists of genes with ease. *Genome Biol*, 4(10):R70, 2003.

[121] Fátima Al-Shahrour, José Carbonell, Pablo Minguez, Stefan Goetz, Ana Conesa, Joaquín Tárraga, Ignacio Medina, Eva Alloza, David Montaner, and Joaquín Dopazo. Babelomics: advanced functional profiling of transcriptomics, proteomics and genomics experiments. *Nucleic Acids Res*, 36(Web Server issue):W341–W346, Jul 2008.

[122] Ignacio Medina, José Carbonell, Luis Pulido, Sara C Madeira, Stefan Goetz, Ana Conesa, Joaquín Tárraga, Alberto Pascual-Montano, Ruben Nogales-Cadenas, Javier Santoyo, Francisco García, Martina Marbà, David Montaner, and Joaquín Dopazo. Babelomics: an integrative platform for the analysis of transcriptomics, proteomics and genomic data with advanced functional profiling. *Nucleic Acids Res*, 38(Web Server issue):W210–W213, Jul 2010.

[123] Juri Reimand, Meelis Kull, Hedi Peterson, Jaanus Hansen, and Jaak Vilo. g:profiler–a web-based toolset for functional profiling of gene lists from large-scale experiments. *Nucleic Acids Res*, 35(Web Server issue):W193–W200, Jul 2007.

[124] Tadashi Imanishi and Hajime Nakaoka. Hyperlink management system and id converter system: enabling maintenance-free hyperlinks among major biological databases. *Nucleic Acids Res*, 37(Web Server issue):W17–W22, Jul 2009.

[125] Gabriel F Berriz and Frederick P Roth. The synergizer service for translating gene, protein and other biological identifiers. *Bioinformatics*, 24(19):2272–2273, Oct 2008.

[126] Daniel Baron, Audrey Bihouee, Raluca Teusan, Emeric Dubois, Frederique Savagner, Marja Steenman, Remi Houlgatte, and Gerard Ramstein. Madgene: retrieval and processing of gene identifier lists for the analysis of heterogeneous microarray datasets. *Bioinformatics*, 27(5):725–6, March 2011.

[127] Andreu Alibés, Patricio Yankilevich, Andrés Cañada, and Ramón Díaz-Uriarte. Idconverter and idclight: conversion and annotation of gene and protein ids. *BMC Bioinformatics*, 8:9, 2007.

[128] Kimberly J Bussey, David Kane, Margot Sunshine, Sudar Narasimhan, Satoshi Nishizuka, William C Reinhold, Barry Zeeberg, Weinstein Ajay, and John N Weinstein. Matchminer: a tool for batch navigation among gene and gene product identifiers. *Genome Biol*, 4(4):R27, 2003.

[129] Cristian I Castillo-Davis and Daniel L Hartl. Genemerge–post-genomic analysis, data mining, and hypothesis testing. *Bioinformatics*, 19(7):891–2, May 2003.

[130] J Tsai, R Sultana, Y Lee, G Pertea, S Karamycheva, V Antonescu, J Cho, B Parvizi, F Cheung, and J Quackenbush. Resourcerer: a database for annotating and linking microarray resources within and across species. *Genome Biol*, 2(11):SOFTWARE0002, 2001.

[131] B Lenhard, W S Hayes, and W W Wasserman. Genelynx: a gene-centric portal to the human genome. *Genome Res*, 11(12):2151–7, December 2001.

[132] Alberto Risueño, Celia Fontanillo, Marcel E Dinger, and Javier De Las Rivas. Gatexplorer: genomic and transcriptomic explorer; mapping expression probes to gene loci, transcripts, exons and ncrnas. *BMC Bioinformatics*, 11:221, 2010.

[133] Ramil N Nurtdinov, Mikhail O Vasiliev, Anna S Ershova, Ilia S Lossev, and Anna S Karyagina. Plandbaffy: probe-level annotation database for affymetrix expression microarrays. *Nucleic Acids Res*, 38(Database issue):D726–D730, Jan 2010.

[134] Pinglang Wang, Fei Ding, Hsienyuan Chiang, Robert C Thompson, Stanley J Watson, and Fan Meng. Probematchdb–a web database for finding equivalent probes across microarray platforms and species. *Bioinformatics*, 18(3):488–9, March 2002.

[135] Eric Jain, Amos Bairoch, Severine Duvaud, Isabelle Phan, Nicole Redaschi, Baris E Suzek, Maria J Martin, Peter McGarvey, and Elisabeth Gasteiger. Infrastructure for the life sciences: design and implementation of the uniprot website. *BMC Bioinformatics*, 10:136, 2009.

[136] Purvesh Khatri, Sorin Draghici, G. Charles Ostermeier, and Stephen A Krawetz. Profiling gene expression using onto-express. *Genomics*, 79(2):266–270, Feb 2002.

[137] Florian Iragne, Aurélien Barré, Nicolas Goffard, and Antoine De Daruvar. Aliasserver: a web server to handle multiple aliases used to refer to proteins. *Bioinformatics*, 20(14):2331–2332, Sep 2004.

[138] Syed Haider, Benoit Ballester, Damian Smedley, Junjun Zhang, Peter Rice, and Arek Kasprzyk. Biomart central portal–unified access to biological data. *Nucleic Acids Res*, 37(Web Server issue):W23–7, July 2009.

[139] Jonathan M Guberman, J Ai, O Arnaiz, Joachim Baran, Andrew Blake, Richard Baldock, Claude Chelala, David Croft, Anthony Cros, Rosalind J Cutts, A Di Genova, Simon Forbes, T Fujisawa, E Gadaleta, D M Goodstein, Gunes Gundem, Bernard Haggarty, Syed Haider, Matthew Hall, Todd Harris, Robin Haw, S Hu, Simon Hubbard, Jack Hsu, Vivek Iyer, Philip Jones, Toshiaki Katayama, R Kinsella, Lei Kong, Daniel Lawson, Yong Liang, Nuria Lopez-Bigas, J Luo, Michael Lush, Jeremy Mason, Francois Moreews, Nelson Ndegwa, Darren Oakley, Christian Perez-Llamas, Michael Primig, Elena Rivkin, S Rosanoff, Rebecca Shepherd, Reinhard Simon, B Skarnes, Damian Smedley, Linda Sperling, William Spooner, Peter Stevenson, Kevin Stone, J Teague, Jun Wang, Jianxin Wang, Brett Whitty, D T Wong, Marie Wong-Erasmus, L Yao, Ken Youens-Clark, Christina Yung, Junjun Zhang, and Arek Kasprzyk. Biomart central portal: an open database network for the biological community. *Database (Oxford)*, 2011:bar041, 2011.

[140] Arek Kasprzyk, Damian Keefe, Damian Smedley, Darin London, William Spooner, Craig Melsopp, Martin Hammond, Philippe Rocca-Serra, Tony Cox, and Ewan Birney. Ensmart: a generic system for fast and flexible access to biological data. *Genome Res*, 14(1):160–169, Jan 2004.

[141] Martijn P van Iersel, Alexander R Pico, Thomas Kelder, Jianjiong Gao, Isaac Ho, Kristina Hanspers, Bruce R Conklin, and Chris T Evelo. The bridgedb framework: standardized access to gene, protein and metabolite identifier mapping services. *BMC Bioinformatics*, 11:5, 2010.

[142] Richard G Cote, Philip Jones, Lennart Martens, Samuel Kerrien, Florian Reisinger, Quan Lin, Rasko Leinonen, Rolf Apweiler, and Henning Hermjakob. The protein identifier cross-referencing (picr) service: reconciling protein identifiers across multiple source databases. *BMC Bioinformatics*, 8:401, 2007.

[143] F. Mohammad, R.M. Flight, B.J. Harrison, J.C. Petruska, and E.C. Rouchka. Interval trees for detection of overlapping genetic entities. In *2011 11th IEEE International Conference on Bioinformatics and Bioengineering*, pages 278–281. IEEE, 2011.

[144] H. Pages, P. Aboyou, and M. Lawrence. Iranges: Infrastructure for manipulating intervals on sequences. *R package version*, 1(6), 2010.

[145] Jeremy Goecks, Anton Nekrutenko, James Taylor, and Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol*, 11(8):R86, 2010.

[146] Daniel Blankenberg, Gregory Von Kuster, Nathaniel Coraor, Guruprasad Ananda, Ross Lazarus, Mary Mangan, Anton Nekrutenko, and James Taylor. Galaxy: a web-based genome analysis tool for experimentalists. *Curr Protoc Mol Biol*, Chapter 19:Unit 19.10.1–21, January 2010.

[147] Belinda Giardine, Cathy Riemer, Ross C Hardison, Richard Burhans, Laura Elnitski, Prachi Shah, Yi Zhang, Daniel Blankenberg, Istvan Albert, James Taylor, Webb Miller, W James Kent, and Anton Nekrutenko. Galaxy: a platform for interactive large-scale genome analysis. *Genome Res*, 15(10):1451–5, October 2005.

[148] Affymetrix hg-u133 plus 2.0 annotation file.

[149] Agilent cgh annotation file.

[150] Chiea-Chuen Khor and Martin L Hibberd. Revealing the molecular signatures of host-pathogen interactions. *Genome Biol*, 12(10):229, October 2011.

[151] Lionel K K Tan, George M Carlone, and Ray Borrow. Advances in the development of vaccines against neisseria meningitidis. *N Engl J Med*, 362(16):1511–20, April 2010.

[152] Cristina Aurrecoechea, John Brestelli, Brian P Brunk, Jennifer Dommer, Steve Fischer, Bindu Gajria, Xin Gao, Alan Gingle, Greg Grant, Omar S Harb, Mark Heiges, Frank Innamorato, John Iodice, Jessica C Kissinger, Eileen Kraemer, Wei Li, John A Miller, Vishal Nayak, Cary Pennington, Deborah F Pinney, David S Roos, Chris Ross, Christian J Stoeckert, Charles Treatman, and Haiming Wang. Plasmodb: a functional genomic database for malaria parasites. *Nucleic Acids Res*, 37(Database issue):D539–43, January 2009.

[153] Dominic P Kwiatkowski. How malaria has affected the human genome and what human genetics can teach us about malaria. *Am J Hum Genet*, 77(2):171–92, August 2005.

[154] Jennifer M Sacheck, Jon-Philippe K Hyatt, Anna Raffaello, R. Thomas Jagoe, Roland R Roy, V. Reggie Edgerton, Stewart H Lecker, and Alfred L Goldberg. Rapid disuse and denervation atrophy involve transcriptional changes similar to those of muscle wasting during systemic diseases. *FASEB J*, 21(1):140–155, Jan 2007.

[155] Stewart H Lecker, R Thomas Jagoe, Alexander Gilbert, Marcelo Gomes, Vickie Baracos, James Bailey, S Russ Price, William E Mitch, and Alfred L Goldberg. Multiple types of skeletal muscle atrophy involve a common program of changes in gene expression. *FASEB J*, 18(1):39–51, January 2004.

[156] R Thomas Jagoe, Stewart H Lecker, Marcelo Gomes, and Alfred L Goldberg. Patterns of gene expression in atrophying skeletal muscles: response to food deprivation. *FASEB J*, 16(13):1697–712, November 2002.

[157] Eric W Sayers, Tanya Barrett, Dennis A Benson, Evan Bolton, Stephen H Bryant, Kathi Canese, Vyacheslav Chetvernin, Deanna M Church, Michael Dicuccio, Scott Federhen, Michael Feolo, Ian M Fingerman, Lewis Y Geer, Wolfgang Helmberg, Yuri Kapustin, Sergey Krasnov, David Landsman, David J Lipman, Zhiyong Lu, Thomas L Madden, Tom Madej, Donna R Maglott, Aron Marchler-Bauer, Vadim Miller, Ilene Karsch-Mizrachi, James Ostell, Anna Panchenko, Lon Phan, Kim D Pruitt, Gregory D Schuler, Edwin Sequeira, Stephen T Sherry, Martin Shumway, Karl Sirotkin, Douglas Slotta, Alexandre Souvorov, Grigory Starchenko, Tatiana A Tatusova, Lukas Wagner, Yanli Wang, W John Wilbur, Eugene Yaschenko, and Jian Ye. Database resources of the national center for biotechnology information. *Nucleic Acids Res*, 40(Database issue):D13–25, January 2012.

[158] Rong Chen, Li Li, and Atul J Butte. Ailun: reannotating gene expression data automatically. *Nat Methods*, 4(11):879, Nov 2007.

[159] Andrea Bisognin, Alessandro Coppe, Francesco Ferrari, Davide Risso, Chiara Romualdi, Silvio Bicciato, and Stefania Bortoluzzi. A-madman: annotation-based microarray data meta-analysis tool. *BMC Bioinformatics*, 10:201, 2009.

[160] Patrick Cahan, Amera M Ahmad, Harry Burke, Sidney Fu, Yinglei Lai, Liliana Florea, Nachiket Dharker, Todd Kobrinski, Prachee Kale, and Timothy A McCaffrey. List of lists-annotated (lola): a database for annotation and comparison of published microarray gene lists. *Gene*, 360(1):78–82, Oct 2005.

[161] Jeffrey C Petruska and Lorne M Mendell. The many functions of nerve growth factor: multiple actions on nociceptors. *Neurosci Lett*, 361(1-3):168–71, may 2004.

[162] Lars J Jensen, Michael Kuhn, Manuel Stark, Samuel Chaffron, Chris Creevey, Jean Muller, Tobias Doerks, Philippe Julien, Alexander Roth, Milan Simonovic, Peer Bork, and Christian von Mering. String 8–a global view on proteins and their functional interactions in 630 organisms. *Nucleic Acids Res*, 37(Database issue):D412–D416, Jan 2009.

[163] Michael Kuhn, Christian von Mering, Monica Campillos, Lars Juhl Jensen, and Peer Bork. Stitch: interaction networks of chemicals and proteins. *Nucleic Acids Res*, 36(Database issue):D684–D688, Jan 2008.

[164] Michael Kuhn, Damian Szklarczyk, Andrea Franceschini, Monica Campillos, Christian von Mering, Lars Juhl Jensen, Andreas Beyer, and Peer Bork. Stitch 2: an interaction network database for small molecules and proteins. *Nucleic Acids Res*, 38(Database issue):D552–D556, Jan 2010.

[165] Jake Yue Chen, SudhaRani Mamidipalli, and Tianxiao Huan. HAPPI: an online database of comprehensive human annotated and predicted protein interactions. *BMC Genomics*, 10 Suppl 1:S16, 2009.

[166] Jingchun Sun, Yan Sun, Guohui Ding, Qi Liu, Chuan Wang, Youyu He, Tieliu Shi, Yixue Li, and Zhongming Zhao. Inpreppi: an integrated evaluation method based on genomic context for predicting protein-protein interactions in prokaryotic genomes. *BMC Bioinformatics*, 8:414, 2007.

[167] Nathan Salomonis, Kristina Hanspers, Alexander C Zambon, Karen Vranizan, Steven C Lawlor, Kam D Dahlquist, Scott W Doniger, Josh Stuart, Bruce R Conklin, and Alexander R Pico. GenMAPP 2: new features and resources for pathway analysis. June 24 2007.

[168] Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. Biogrid: a general repository for interaction datasets. *Nucleic Acids Res*, 34(Database issue):D535–D539, Jan 2006.

[169] Arnaud Ceol, Andrew Chatr Aryamontri, Luana Licata, Daniele Peluso, Leonardo Briganti, Livia Perfetto, Luisa Castagnoli, and Gianni Cesareni. Mint, the molecular interaction database: 2009 update. *Nucleic Acids Res*, 38(Database issue):D532–D539, Jan 2010.

[170] B. Aranda, P. Achuthan, Y. Alam-Faruque, I. Armean, A. Bridge, C. Derow, M. Feuermann, A. T. Ghanbarian, S. Kerrien, J. Khadake, J. Kerssemakers, C. Leroy, M. Menden, M. Michaut, L. Montecchi-Palazzi, S. N. Neuhauser, S. Orchard, V. Perreau, B. Roechert, K. van Eijk, and H. Hermjakob. The intact molecular interaction database in 2010. *Nucleic Acids Res*, 38(Database issue):D525–D531, Jan 2010.

[171] Tomas Klingstrom and Dariusz Plewczynski. Protein-protein interaction and pathway databases, a graphical review. *Brief Bioinform*, sep 2010.

[172] Sara Mostafavi, Debajyoti Ray, David Warde-Farley, Chris Grouios, and Quaid Morris. Genemania: a real-time multiple association network integration algorithm for predicting gene function. *Genome Biol*, 9 Suppl 1:S4, 2008.

[173] Sara Mostafavi and Quaid Morris. Fast integration of heterogeneous data sources for predicting gene function with limited annotation. *Bioinformatics*, 26(14):1759–65, jul 2010.

[174] Adi L. Tarca, Sorin Draghici, Purvesh Khatri, Sonia S. Hassan, Pooja Mittal, Jung sun Kim, Chong Jai Kim, Juan Pedro Kusanovic, and Roberto Romero. A novel signaling pathway impact analysis. *Bioinformatics*, 25(1):75–82, 2009.

[175] Daniel C Kirouac, Caryn Ito, Elizabeth Csaszar, Aline Roch, Mei Yu, Edward A Sykes, Gary D Bader, and Peter W Zandstra. Dynamic interaction networks in a hierarchically organized tissue. *Mol Syst Biol*, 6:417, oct 2010.

[176] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res*, 13(11):2498–504, November 2003.

[177] Melissa S Cline, Michael Smoot, Ethan Cerami, Allan Kuchinsky, Nerius Landys, Chris Workman, Rowan Christmas, Iliana Avila-Campilo, Michael Creech, Benjamin Gross, Kristina Hanspers, Ruth Isserlin, Ryan Kelley, Sarah Killcoyne, Samad Lotia, Steven Maere, John Morris, Keiichiro Ono, Vuk Pavlovic, Alexander R Pico, Aditya Vailaya, Peng-Liang Wang, Annette Adler, Bruce R Conklin, Leroy Hood, Martin Kuiper, Chris Sander, Ilya Schmulevich, Benno Schwikowski, Guy J Warner, Trey Ideker, and Gary D Bader. Integration of biological networks and gene expression data using cytoscape. *Nat Protoc*, 2(10):2366–82, 2007.

[178] Gordon K. Smyth and Gordon K. Smyth. Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Stat. Appl. Genet. Mol. Biol*, 2004.

[179] Robert C Gentleman, Vincent J Carey, Douglas M Bates, Ben Bolstad, Marcel Dettling, Sandrine Dudoit, Byron Ellis, Laurent Gautier, Yongchao Ge, Jeff Gentry, Kurt Hornik, Torsten Hothorn, Wolfgang Huber, Stefano Iacus, Rafael Irizarry, Friedrich Leisch, Cheng Li, Martin Maechler, Anthony J Rossini, Gunther Sawitzki, Colin Smith, Gordon Smyth, Luke Tierney, Jean Y H Yang, and Jianhua Zhang. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol*, 5(10):R80, 2004.

Appendix A

SUPPLEMENTARY TABLES

A.1 Entrez IDs converted to GeneSymbol

Table A.1: Entrez IDs converted to gene symbols uniquely by AbsIDconvert.

EntrezID	Gene symbol (NCBI)	Gene type (NCBI)	Gene symbol (AbsIDconvert)
100505607	LOC100505607	miscRNA	LOC100505607
100505645	LOC100505645	miscRNA	LOC100505645
100505920	LOC100505920	miscRNA	LOC100505920
100505938	LOC100505938	miscRNA	LOC100505938
100505950	LOC100505950	miscRNA	LOC100505950
100506044	LOC100506044	miscRNA	LOC100506044
100506122	LOC100506122	miscRNA	LOC100506122
100506123	LOC100506123	miscRNA	LOC100506123
100506130	LOC100506130	miscRNA	LOC100506130
100506158	LOC100506158	miscRNA	LOC100506158
100506192	LOC100506192	miscRNA	LOC100506192
100506272	LOC100506272	miscRNA	LOC100506272
100506329	LOC100506329	miscRNA	LOC100506329
100506351	LOC100506351	miscRNA	LOC100506351
100506452	LOC100506452	miscRNA	LOC100506452
100506609	LOC100506609	miscRNA	LOC100506609
100506695	LOC100506695	miscRNA	PHF21B, LOC100506695
100506837	LOC100506837	miscRNA	LOC100506837
100507153	LOC100507153	miscRNA	LOC100507153
100507205	LOC100507205	miscRNA	LOC100507205
100507389	LOC100507389	miscRNA	LOC100507389
100507581	LOC100507581	miscRNA	LOC100507581
100507615	LOC100507615	miscRNA	LOC100507615
100507672	LOC100507672	miscRNA	PPARD, LOC100507672
100529145	TEN1-CDK3	miscRNA	C17ORF106-CDK3, TEN1, CDK3
100529211	C17orf61-PLSCR3	miscRNA	C17ORF61-PLSCR3, PLSCR3, C17ORF61

Continued on next page...

Table A.1 – continued from previous page

Entrez ID	Gene symbol (NCBI)	Gene type (NCBI)	Gene symbol (AbsIDconvert)
100630923	LOC100630923	miscRNA	LOC100630923, LOC100289561, PRKRIP1
100652780	LOC100652780	other	LOC100652780
100329135	LOC100329135	protein-coding	LOC100329135
100505549	LOC100505549	protein-coding	LOC100505549
100507421	LOC100507421	protein-coding	LOC100507421, LOC100130169
100652826	LOC100652826	protein-coding	LOC100652826
100303743	LOC100303743	pseudo	WWOX, LOC100303743
100418753	LOC100418753	pseudo	LOC100418753
100418754	LOC100418754	pseudo	LOC100418754
100418951	LOC100418951	pseudo	LOC100418951
100418955	LOC100418955	pseudo	LOC100418955
100419014	LOC100419014	pseudo	LOC100419014
100419017	LOC100419017	pseudo	LOC100419017
100419108	LOC100419108	pseudo	RBMS1, LOC100419108
100419553	LOC100419553	pseudo	LOC100419553
100419621	LOC100419621	pseudo	TADA2A, LOC100419621
100419694	LOC100419694	pseudo	LOC100419694
100419779	LOC100419779	pseudo	LOC100419779
100419814	LOC100419814	pseudo	CUL3, LOC100419814
100419892	LOC100419892	pseudo	LOC100419892
100419951	LOC100419951	pseudo	LOC100419951
100419986	LOC100419986	pseudo	LOC100419986
100420064	LOC100420064	pseudo	LOC100420064
100420177	LOC100420177	pseudo	LOC100420177
100420305	LOC100420305	pseudo	LOC100420305
100420358	LOC100420358	pseudo	LOC100420358
100420741	LOC100420741	pseudo	LOC100420741
100420886	LOC100420886	pseudo	WDR35, LOC100420886
100420949	LOC100420949	pseudo	LOC100287225, LOC100420949
100421028	LOC100421028	pseudo	LOC100421028
100421121	LOC100421121	pseudo	LOC100421121
100421437	LOC100421437	pseudo	ZNF148, LOC100421437
100421471	LOC100421471	pseudo	LOC100421471
100421494	LOC100421494	pseudo	LOC100421494
100421695	LOC100421695	pseudo	LOC100421695
100422265	LOC100422265	pseudo	LOC100422265
100422284	LOC100422284	pseudo	LOC100422284
100422299	LOC100422299	pseudo	LOC100422299
100422510	LOC100422510	pseudo	LOC100422510

Continued on next page...

Table A.1 – continued from previous page

Entrez ID	Gene symbol (NCBI)	Gene type (NCBI)	Gene symbol (AbsIDconvert)
100422524	LOC100422524	pseudo	MLLT10, LOC100422524
100422671	LOC100422671	pseudo	LOC100422671
100500934	LOC100500934	pseudo	LOC100500934
100507388	LOC100507388	pseudo	LOC100505505, LOC100507388
100507595	LOC100507595	pseudo	LRRC33, LOC100507595
100526736	LOC100526736	pseudo	LOC100526736
100533622	LOC100533622	pseudo	LOC100533622
100533658	LOC100533658	pseudo	LOC100533658
100533661	LOC100533661	pseudo	LOC100533661
100533663	LOC100533663	pseudo	LOC100533663
100533732	LOC100533732	pseudo	LOC100533732
100533846	LOC100533846	pseudo	LOC100533846
100631258	LOC100631258	pseudo	NELL1, LOC100631258
100652752	LOC100652752	pseudo	LOC100652752
100652792	LOC100652792	pseudo	LOC100652792
100505530	FLJ45825	unknown	LOC100505530
100506282	LOC100506282	unknown	LOC100506282
100506342	LOC100506342	unknown	LOC100506342

Table A.2: Entrez IDs converted to gene symbol by HMS & ID, DAVID and MADGene missed by AbsIDConvert.

EntrezID	Gene symbol (NCBI)	Gene type (NCBI)	NCBI annotation	HMS & ID	DAVID	MADGene
286750	–	–	replaced withGene ID: 9414	–	DFNA51	DFNA51
5374	–	–	replaced withGene ID: 390831	–	PMM2P1	PMM2P1
728795	–	–	replaced withGene ID: 100420746	–	LOC728795	hCG_1644355
493620	–	–	replaced withGene ID: 100418887	–	TAGLN2P1	TAGLN2P1
645128	–	–	replaced withGene ID: 100288486	–	LOC645128	LOC645128
100507343	CPB2-AS1	miscRNA	Not on current assembly	CPB2-AS1	–	–
100129836	COL4A2-AS2	miscRNA	Not on current assembly	COL4A2-AS2	LOC100129836	LOC100129836
57234	FAM91A2	miscRNA	Not on current assembly	FAM91A2	FAM91A2	FAM91A2
693204	MIR619	miscRNA	Not on current assembly	MIR619	MIR619	MIR619
729634	KRT18P26	pseudo	Not on current assembly	KRT18P26	KRT18P19, KRT18, KRT18P26	KRT18P26
100189058	TRNAQ9	tRNA	Not on current assembly	TRNAQ9	TRNAQ9	TRNAQ9
283486	LINC00567	unknown	Not on current assembly	LINC00567	LOC283486	LOC283486
100131497	LOC100131497	miscRNA	Not on current assembly	–	LOC100131497	LOC100131497
401021	LOC401021	miscRNA	Not on current assembly	–	LOC401021	LOC401021
729683	LOC729683	miscRNA	Not on current assembly	–	LOC729683	LOC729683
100288085	DYZ1L5	other	Not on current assembly	–	LOC100288085	LOC100288085
100131004	LOC100131004	protein-coding	Not on current assembly	–	LOC100131004	LOC100131004
100131310	LOC100131310	protein-coding	Not on current assembly	–	LOC100131310	LOC100131310
100132365	LOC100132365	protein-coding	Not on current assembly	–	LOC100132365	LOC100132365
100286906	LOC100286906	protein-coding	Not on current assembly	–	LOC100286906	LOC100286906
100289058	LOC100289058	protein-coding	Not on current assembly	–	LOC100289058	LOC100289058

Continued on next page...

Table A.2 – continued from previous page

Entrez ID	Gene symbol (NCBI)	Gene type (NCBI)	NCBI annotation	HMS & ID	DAVID	MADGene
653541	LOC653541	protein-coding	Not on current assembly	–	LOC399839, HPX-2, LOC728410, LOC653541, LOC653548, LOC653544, LOC653543, LOC653545, LOC440014, LOC440013, LOC441056, LOC728022, DUX4, LOC652119, LOC440017	LOC653541
727961	LOC727961	protein-coding	Not on current assembly	–	LOC727961	hCG_1776047
100128019	LOC100128019	unknown	Not on current assembly	–	LOC100128019	LOC100128019
100129894	LOC100129894	unknown	Not on current assembly	–	LOC100129894	LOC100129894
100131043	LOC100131043	unknown	Not on current assembly	–	LOC100131043	LOC100131043
100288869	LOC100288869	unknown	Not on current assembly	–	LOC100288869	LOC100288869
400943	LOC400943	unknown	Not on current assembly	–	UNQ5830	UNQ5830
440479	FLJ34223	unknown	Not on current assembly	–	FLJ34223	FLJ34223
645895	LOC645895	unknown	Not on current assembly	–	LOC645895	LOC645895
645967	LOC645967	unknown	Not on current assembly	–	LOC645967	LOC645967
648149	LOC648149	unknown	Not on current assembly	–	LOC648149	LOC648149
727799	LOC727799	unknown	Not on current assembly	–	LOC727799	LOC727799
693128	MIR548B	miscRNA	Not annot. on reference assembly	MIR548B	MIR548B	MIR548B
28332	IGHD2OR15-2B	other	Not annot. on reference assembly	IGHD2OR15-2B	IGHD2OR15-2B	IGHD2OR15-2B
3506	IGHJ@	other	Not annot. on reference assembly	IGHJ@	IGHJ@	IGHJ@

Continued on next page...

Table A.2 – continued from previous page

Entrez ID	Gene symbol (NCBI)	Gene type (NCBI)	NCBI annotation	HMS & ID	DAVID	MADGene
100313795	PIRC73	other	Not annot. on reference assembly	PIRC73	–	–
100313815	PIRC54	other	Not annot. on reference assembly	PIRC54	–	–
28301	IGHV3OR16-15	pseudo	Not annot. on reference assembly	IGHV3OR16-15	IGHV3OR16-15	IGHV3OR16-15
28854	IGKV3OR2-5	pseudo	Not annot. on reference assembly	IGKV3OR2-5	IGKV3OR2-5	IGKV3OR2-5
28861	IGKV2OR2-1	pseudo	Not annot. on reference assembly	IGKV2OR2-1	IGKV2OR2-1	IGKV2OR2-1
4699	NDUFA5P1	pseudo	Not annot. on reference assembly	NDUFA5P1	NDUFA5P1	NDUFA5P1
654813	P2RY10P1	pseudo	Not annot. on reference assembly	P2RY10P1	P2RY10P1	P2RY10P1
100189517	TRNAN35	tRNA	Not annot. on reference assembly	TRNAN35	TRNAN35P	TRNAN35P
25784	DGCR12	unknown	Not annot. on reference assembly	DGCR12	DGCR12	DGCR12
3405	IDDM6	unknown	Not annot. on reference assembly	IDDM6	IDDM6	IDDM6
4375	MRX11	unknown	Not annot. on reference assembly	MRX11	MRX11	MRX11
554188	FCMTE2	unknown	Not annot. on reference assembly	FCMTE2	FCMTE2	FCMTE2
594832	MYP11	unknown	Not annot. on reference assembly	MYP11	MYP11	MYP11
6893	TAPVR1	unknown	Not annot. on reference assembly	TAPVR1	TAPVR1	TAPVR1
7889	PSORS3	unknown	Not annot. on reference assembly	PSORS3	PSORS3	PSORS3
882	CCAL1	unknown	Not annot. on reference assembly	CCAL1	CCAL1	CCAL1
89760	MRX75	unknown	Not annot. on reference assembly	MRX75	MRX75	MRX75
100302562	MENAQ2	unknown	Not annot. on reference assembly	–	–	MENAQ2
400579	FLJ35934	miscRNA	Not annot. on reference assembly	–	FLJ35934	FLJ35934
286009	LOC286009	other	Not annot. on reference assembly	–	LOC286009	tcag7.929
387281	LCRB	other	Not annot. on reference assembly	–	LCRB	LCRB
402469	LOC402469	other	Not annot. on reference assembly	–	LOC402469	tcag7.1056
26101	DKFZP564M1462	protein-coding	Not annot. on reference assembly	–	DKFZP564M1462	DKFZP564M1462
283911	LOC283911	protein-coding	Not annot. on reference assembly	–	LOC283911	LOC283911
55547	HAB1	protein-coding	Not annot. on reference assembly	–	HAB1	HAB1
653486	LOC653486	protein-coding	Not annot. on reference assembly	–	LOC653486	hCG_1741344
100133452	LOC100133452	pseudo	Not annot. on reference assembly	–	LOC100133452	LOC100133452
100292981	LOC100292981	pseudo	Not annot. on reference assembly	–	LOC100292981	LOC100292981
100294336	LOC100294336	pseudo	Not annot. on reference assembly	–	LOC100294336	LOC100294336
647349	LOC647349	pseudo	Not annot. on reference assembly	–	LOC647349	LOC647349

Continued on next page...

Table A.2 – continued from previous page

Entrez ID	Gene symbol (NCBI)	Gene type (NCBI)	NCBI annotation	HMS & ID	DAVID	MADGene
652073	LOC652073	pseudo	Not annot. on reference assembly	-	LOC652073	LOC652073
730535	LOC730535	pseudo	Not annot. on reference assembly	–	LOC730535	LOC730535
100049541	STUT1	unknown	Not annot. on reference assembly	–	STUT1	STUT1
100134822	LOC100134822	unknown	Not annot. on reference assembly	–	LOC100134822	LOC100134822
100188748	STHAG5	unknown	Not annot. on reference assembly	–	STHAG5	STHAG5
100188844	MAFD6	unknown	Not annot. on reference assembly	–	MAFD6	MAFD6
100188852	SHEP8	unknown	Not annot. on reference assembly	–	SHEP8	SHEP8
100188853	BMND7	unknown	Not annot. on reference assembly	–	BMND7	BMND7
100190985	IS5	unknown	Not annot. on reference assembly	–	IS5	IS5
100271922	BFIC4	unknown	Not annot. on reference assembly	–	BFIC4	BFIC4
100293044	LOC100293044	unknown	Not annot. on reference assembly	–	LOC100293044	LOC100293044
338030	GLM1	unknown	Not annot. on reference assembly	–	GLM1	GLM1
414058	ACRPV	unknown	Not annot. on reference assembly	–	ACRPV	ACRPV
474225	RA5	unknown	Not annot. on reference assembly	–	RA5	RA5
474261	BP15	unknown	Not annot. on reference assembly	–	BP15	BP15
474285	OA1	unknown	Not annot. on reference assembly	–	OA1	GPR143
474294	BW19	unknown	Not annot. on reference assembly	–	BW19	BW19
474295	BW20	unknown	Not annot. on reference assembly	–	BW20	BW20
50989	HMSNO	unknown	Not annot. on reference assembly	–	HMSNO	HMSNO
544599	AASTH45	unknown	Not annot. on reference assembly	–	AASTH45	AASTH45
544616	COHEN2	unknown	Not annot. on reference assembly	–	COHEN2	COHEN2
780911	TQDS	unknown	Not annot. on reference assembly	–	TQDS	TQDS
780925	CHMRQ	unknown	Not annot. on reference assembly	–	CHMRQ	CHMRQ
7834	PCAP	unknown	Not annot. on reference assembly	–	PCAP	PCAP
8008	BDMF	unknown	Not annot. on reference assembly	–	BDMF	BDMF
8041	TP250	unknown	Not annot. on reference assembly	–	TP250	TP250
8173	CNSN	unknown	Not annot. on reference assembly	–	CNSN	CNSN
8205	TAM	unknown	Not annot. on reference assembly	–	TAM	TAM

A.2 Entrez IDs converted to RefSeq

Table A.3: Entrez IDs converted to Refseq by MADGene missed by AbsIDConvert.

EntrezID	RefSeq (NCBI)	MADGene	DAVID	Onto-Translate
6080	NR_002907	NR_002907	NR_004385, NR_004406, NR_004404, NR_002907, NR_004386	NR_002907
26822	NR_000022	NR_000022	NR_001452, NR_001454, NR_001453, NR_003125, NR_000022	NR_000022
100302146	NR_031634	NR_031634	-	-
100302193	NR_031656	NR_031656	-	-
100302167	NR_031629	NR_031629	NR_031629	-

Table A.4: Genomic intervals found by AbsIDconvert for the five unmapped Entrez IDs found by MADGene.

Entrez ID	chromosome	start	end	width	strand
6080	chr1	28833877	28834083	207	+
26822	chr11	17096200	17096291	92	-
100302146	chr20	49231173	49231322	150	-
100302193	chr4	102251459	102251571	113	-
100302167	chr9	69002239	69002321	83	-

Table A.5: Entrez IDs to RefSeq conversion by DAVID, with missing annotation from NCBI.

EntrezID	DAVID	NCBI Entrez annotation for DAVID RefSeq
100129552	NM_001029	6231
285176	NM_006013, NR_026898	6134
388474	NM_000972	6130
440991	NM_001005	6188
642538	NM_006333, NM_173177	10438
642585	NM_003374	7416
644634	NR_027002	388692
646050	NM_022831	64853
653252	NM_006327	100287932
727828	NM_001164397	642446
727984	NM_001035006, NM_000985	6139, 6140
728513	NM_032882	84968
728533	NM_014761	9798
728698	NM_001416, NR_002912	1973
728953	NM_001022	6223
728970	NM_025113	80183
729163	NM_001444	2171
729458	NM_144614	125997
729992	NM_003932	6767
81458	NM_001001824	403239

Table A.6: Entrez IDs converted to RefSeq IDs exclusively by AbsIDconvert.

EntrezID	RefSeq	AbsIDconvert
81104	-	NR_015416
100505905	-	NM_001256876, NM_001256877
400433	-	NR_033787
100131381	-	NR_029697
100652874	-	NR_046251, NR_046252, NR_046253, NR_046254, NR_046255
100463488	NM_001190708	NM_001190708
100271846	NM_001191055	NM_001191055
642612	NM_001195234	NM_001195234
100507421	NM_001195278	NM_001195278
100287466	NM_001242319	NM_001242319, NM_032882
100313837	NR_031576	NR_031576
100329109	NR_033248	NR_033248
647135	NR_034178	NR_034178
100422851	NR_036200	NR_036200
100422860	NR_036251	NR_036251
284648	NR_036490	NR_036490
100289373	NR_036531, NR_036532	NR_036531, NR_036532
100500878	NR_037450	NR_037450
100506548	NR_037665	NR_037665
100359394	NR_037842	NR_037842
100507582	NR_037903	NR_037903
100505687	NR_038301, NR_038302	NR_038301, NR_038302
554206	NR_038379	NR_038379
729444	NR_038388	NR_038388
100129464	NR_038428	NR_038428
253962	NR_038439	NR_038439
147093	NR_038442	NR_038442
284865	NR_038460	NR_038460
100507401	NR_038909	NR_038909
100506241	NR_038954	NR_038954
100616164	NR_039616	NR_039616
100616469	NR_039627	NR_039627
100616499	NR_039634	NR_039634, NR_039636
100616399	NR_039635	NR_039635
100616315	NR_039942	NR_039942
284395	NR_040029	NR_040029
81343	NR_045005	NR_045005
440519	NR_045525	NR_045525

Table A.7: Genomic intervals for AbsIDconvert Entrez to RefSeq conversion.

chromosome	start	end	width	strand	name	chromosome	start	end	width	strand	name
chr15	22332368	22333348	981	+	81104	chr15	22332387	22332775	389	+	NR_015416
chr9	540552	549535	8984	+	100505905	chr9	540507	540666	160	+	NM_001256876
chr9	540552	549535	8984	+	100505905	chr9	549106	549720	615	+	NM_001256877
chr15	85046982	85049663	2682	+	400433	chr15	85046594	85050248	3655	+	NR_033787
chr11	43600529	43606151	5623	+	100131381	chr11	43602943	43603032	90	+	NR_029697
chr3	14961857	14989931	28075	-	100652874	chr3	14984285	14987660	3376	-	NR_046252
chr3	14961857	14989931	28075	-	100652874	chr3	14984285	14987660	3376	-	NR_046253
chr3	14961857	14989931	28075	-	100652874	chr3	14984285	14987660	3376	-	NR_046254
chr3	14961857	14989931	28075	-	100652874	chr3	14984285	14987660	3376	-	NR_046255
chr3	14961857	14989931	28075	-	100652874	chr3	14984285	14987660	3376	-	NR_046251
chr3	14961857	14989931	28075	-	100652874	chr3	14988593	14989011	419	-	NR_046252
chr3	14961857	14989931	28075	-	100652874	chr3	14988616	14989011	396	-	NR_046253
chr3	14961857	14989931	28075	-	100652874	chr3	14988620	14989011	392	-	NR_046254
chr3	14961857	14989931	28075	-	100652874	chr3	14989245	14989399	155	-	NR_046255
chr3	14961857	14989931	28075	-	100652874	chr3	14989519	14989947	429	-	NR_046251

INDEX

CURRICULUM VITAE

Fahim Mohammad
Computer Engineering and Computer Science Department
University of Louisville
Louisville, KY 40292
Email: fahim.md@gmail.com

Objective

Seeking a position as a machine learning and bioinformatics researcher in a challenging environment where my educational background, diverse skills and passion for research can be effectively utilized for positive growth of the organization.

Education

- Ph.D., Computer Science and Engineering
  University of Louisville, Louisville, KY, USA
  Dissertation Title: A systems–based approach for detecting molecular interactions across tissues.
  Advisor: Eric C. Rouchka, *D.Sc.*
  CGPA: 3.92/4.0; Expected graduation: July 2012.

- M.Tech. (Information Technology), 2004–2007
  GGS Indraprastha University, Delhi, India
  Master's project title: A text mining approach for Ontology learning.
  Secured first division (76.35%, Distinction)

- B. Tech. (Computer Science and Engineering), 1997–2001
  Aligarh Muslim University, Aligarh, India
  Bachelor's Project Title: Software quality measurement tool
  Secured first division (8.4 CGPA out of 10.0).

Experience

- Research: Four years as graduate research assistant at University of Louisville.
- Teaching: Six years experience as Lecturer in Computer Science and Engineering.

Computer skills

Programming Languages: R, C, C++, MATLAB, Perl, JAVA, CUDA
Database packages: MySQL, SQL server, Oracle, SPARQL
Scripting language: PHP, HTML, DHTML, JavaScript, AJAX, CSS, XML.

Operating Systems: Linux and Windows
Methodology: Structured and Object Oriented based Design
Statistical tools: R, Weka, Matlab, SAS, SPSS
Other tools: Protg, RDF, Stanford parser

Software tools developed

1. "AbsIDconvert: An Absolute approach fot converting Genetic Identifiers at Different Granularitiess", Available online at `http://bioinformatics.louisville.edu/abid/`. As of June 1^{st}, 2012, AbsIDconvert provides analysis support for 53 organisms.

2. "AbsIDconvert for bacterial genomes", available online at `http://bioinformatics.louisville.edu/babid/`. It provides support for 1497 bacterial strains.

Journal publications

1. Mohammad F, Flight RM,Harrison BJ, Petruska JC,Rouchka EC."AbsIDconvert: An Absolute approach fot converting Genetic Identifiers at Different Granularitiess", BMC Bioinformatics, (Submitted).

Refereed conference publications

1. Mohammad F, Flight RM,Harrison BJ, Petruska JC,Rouchka EC.(2011), "A Heuristic Algorithm for Detecting Intercellular Interactions Among Proteins",Proceedings of the 11^{th} IEEE International Conference on Bioinformatics and Bioengineering (BIBE 2011), pp. 1-8. October 24^{th} - 26^{th}, 2011, Taichung, Taiwan.

2. Mohammad F, Flight RM,Harrison BJ, Petruska JC,Rouchka EC.(2011), "Interval Trees for Detection of Overlapping Genetic Entities",Proceedings of the 11^{th} IEEE International Conference on Bioinformatics and Bioengineering (BIBE 2011), pp. 278-281. October 24^{th} - 26^{th}, 2011, Taichung, Taiwan.

3. Li D, Fahim M, Rouchka EC, "A Bayesian nonparametric model for joint relation integration and domain clustering", In Proceedings of the Ninth International Conference on Machine Learning and Applications (ICMLA 2010). December 12^{th} - 14^{th}, 2010, Washington, DC.

Conference publication

1. Fahim M, "Text mining approach for ontology learning", National conference on Emerging trends and application in computer science, NCETA, April 13^{th}, held at Ajmer, India.

2. Fahim M and Tanvir Ahmad, "DNA Computing - Evolution of a new computing framework", National conference on energy, communication and computer systems, NC-ECC February 2^{nd} - 4^{th}, 2006, held at MAIT, New Delhi, India.

Journal abstract

1. Bing Wang, Fahim Mohammad, Jun Zhang, Xinmin Yiin, Eric rouchka and Xiang Zhang, "Statistical Analysis of multiple significance test methods for differential proteomics", BMC Bioinformatics, 11 (Suppl 4): P30, 2010.

Poster presented

1. "AbsIDconvert : An absolute approach for converting genetic identifiers at different granularities", Institute for Pure and Applied Mathematics (IPAM), University of California, Los Angeles, USA October 3^{rd} - 6^{th}, 2011.

2. "Interval Trees for Detection of Overlapping Genetic Entities", 11^{th} IEEE International Conference on Bioinformatics and Bioengineering, Taiwan, October 24^{th} , 2011.

3. "A novel approach for finding statistical significance in differential proteomics", E-Expo, University of Louisville, March 6^{th}, 2010.

Research talks

1. "AbsIDconvert : An absolute approach for converting genetic identifiers at different granularities", 11^{th} annual UT-ORNL-KBRIN Bioinformatics summit 2011, Louisville, March 30^{th}, 2012

2. "A Heuristic Algorithm for Detecting and Predicting Intercellular interaction", 11^{th} IEEE International Conference on Bioinformatics and Bioengineering, Taiwan, October 24^{th} , 2011

3. "A systems based approach to find protein interaction across tissues", 10^{th} annual UT-ORNL-KBRIN Bioinformatics summit 2011, Memphis, April 1^{st}, 2011

4. "Meta-analysis for studying gene expression data", Bioinfomatics journal club, University of Louisville, November 17^{th}, 2010.

5. "A Systems-based approach for meta-analysis", Graduate Research Symposium, University of Louisville, March 5^{th}, 2010.