PAIRED END TRANSCRIPTOME ASSEMBLY AND GENOMIC VARIANTS MANAGEMENT FOR NEXT GENERATION SEQUENCING DATA

CAI SHAOJIANG

(B.ENG., RENMIN UNIVERSITY OF CHINA)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE (BY RESEARCH)

SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF SINGAPORE

2014

DECLARATION

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Cai Shaojiang

16th May 2014

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my supervisors Prof. Danny Poo and Prof. Wing-Kin Sung for the continuous support of my study and research, for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis. I appreciate the unconditional support from Prof. Sung for valuable guidance and inspiration on the project PETA.

Besides my supervisors, I would like to thank the rest of my thesis committee: Prof. Chan Hock Chuan, Prof. Wong Limsoon and Prof. Teo Yong Meng, for their encouragement, insightful comments, and hard questions.

My sincere thanks also goes to Dr. James Mah, who brought me to the exciting world of Bioinformatics. I would never forget that he briefed me the foundations of SNP research, opening the door to an exciting world for me. Also I would like to thank Pramila from GIS, who gave insightful comments for my research.

I thank my lovely friends in Information Systems Department: Wang Qingliang, Luo Cheng, Cheng Yihong, Feng Yuanyue, Lek Hsiang Hui, Chen Qing, Li Zhuolun and Zhou Hufeng, for the happiest time in basketball fields and sleepless nights before deadlines. Without them, the research life would not be so colorful.

Last but not the least, I express my deepest love to my family: my parents Cai Liansong and Yang Axian, and my sister Cai Qinxiang, for supporting me spiritually throughout my life. And much love goes to my wife Xu Yiling, who is always right there supporting and encouraging me.

Table of Contents

SUMMARY

Next generation sequencing (NGS) techniques accelerate the genomic and transcriptomic studies by providing high throughput, low cost sequencing. However, the overwhelming sequencing data poses demanding challenges for data analysis and management. In this dissertation, we discuss about two methods that process large-scale NGS data, i.e., PETA (Paired End Transcriptome Assembler) and UASIS (Universal Automated SNP Identification System). Both of them are practical and powerful tools to provide enhanced NGS services.

The first study deals with the problem of *de novo* transcriptome assembly. Overwhelming RNA-seq reads, which are often very short, pose a significant informatics challenge to reconstruct the full picture of transcriptome, especially when a high-quality reference genome sequence is not available to serve as a guide. Although the third-generation sequencing is able to provide full-length cDNA reads, we observe that they still suffer from high error rates and low abundance. Accurate and efficient assemblers are still essential for transcriptome analysis.

Nowadays, transcriptome assembly generally follows the development of genome assembly, in which coverage information is widely and reliably used for contig extension, error detection and correction. However, highly fluctuated coverage in RNA-seq libraries makes genome assemblers inadequate to handle alternative splicing patterns. The data structure *de Bruijn* graph is widely used in transcriptome assembly projects. Since the reads are chopped into short k-mers and the paired-end information is lost, current assemblers do not fully utilize the information extracted from the datasets. They usually map the paired-end reads back to the graph structure at a later stage. But the mapping task itself is difficult especially when the graph is complex.

We develop a new *de novo* transcriptome assembler called PETA (Paired End Transcriptome Assembler). We claim that the full utilization of raw reads and paired-end information is able to construct a cleaner splicing graph and generate more accurate and reliable transcriptome. We follow the classical overlap-layout-consensus scheme and use the full reads for extension, which are usually much longer than k-mers and hence more reliable. Paired-end information is widely used for contig extension, validation and graph processing. It is especially good at assembling low coverage regions where k-mer based methods may fail. Our experiments show that PETA outperforms other state-of-art *de novo* assemblers.

High-quality transcriptomes help researchers to do thorough Genome-Wide Association Studies (GWAS), which typically focus on associations between Single Nucleotide Polymorphism (SNPs) and traits of major diseases, such as cancer. RNA-seq has been applied to identify the isoforms that are differently expressed between the normal and tumor samples. More researchers are utilizing RNA-seq techniques to detect SNPs in the transcriptomes. For all of these GWAS applications, PETA serves as a fundamental component, from which other analysis can be performed. However, we have observed some problems in the management of SNPs.

As NGS techniques become popular, overwhelming data introduces chaos for efficient management of genomic variants, especially SNPs. There has been an explosion of data available for public use. SNP databases such as dbSNP, GWAS (formerly HGVbaseG2P), HapMap and JSNP have collected millions of records. But the same SNP may be assigned different identities in these databases. Our second study proposes a novel nomenclature to achieve better management of SNPs on human genome. We develop a SNP nomenclature centralization application called UASIS (Universal Automated SNP Identification System) to resolve the heterogeneous representations of SNPs.

UASIS is a web application for SNP nomenclature standardization and translation. Three utilities are available. They are UASIS Aligner, Universal SNP Name Generator and SNP Name Mapper. UASIS maps SNPs from different databases, including dbSNP, GWAS, HapMap and

JSNP etc., into an uniform view efficiently using a proposed universal nomenclature and state-of-art alignment algorithms.

The thesis contributes to the bioinformatics community by providing two powerful tools, PETA and UASIS, to interpret and analyze large scale of Next Generation Sequencing data. They serve as fundamental components to provide accurate transcriptomes and better data management for related studies like gene expression analysis and GWAS.

List of Tables

List of Figures

List of Algorithms

Glossary

RNA Ribonucleic acid, which carries the genetic information that directs the synthesis of proteins.

mRNA Messenger RNA. An RNA product that is transcribed from the DNA and ultimately transported to a ribosome where it is translated into protein.

cDNA Complementary DNA. DNA synthesized from a messenger RNA (mRNA) template in a reaction catalyzed by the enzyme reverse transcriptase and the enzyme DNA polymerase.

NGS Next Generation Sequencing. A new set of technologies producing thousands or millions of sequences concurrently.

RNA-seq (or mRNA-seq) The most popular protocol for measuring RNA levels using NGS technologies.

Read A sequence of DNA bases generated by a sequencer.

Mate In a paired-end RNA-seq library, the two in-paired reads are called the mate (or mate read) of each other.

Insert size The distance between the paired reads on the sequenced DNA or cDNA.

***de novo* assembly** Constructing a transcriptome in the absence of an assembled genome sequence for the organism.

EST Expressed Sequence Tag, a short subsequence of a cDNA sequence to identify genes.

PETA Paired End Transcriptome Assembler. It is the name of our assembler.

K-MER A length-k DNA nucleotide sequence.

TEMPLATE A sequence of nucleotide characters. It grows longer and longer when PETA runs.

JUNCTION A connection between two templates.

TAIL A subsequence located at either end of a template. Its length is defined by users and must be shorter than the read length. It is used to extend templates.

SPLICING GRAPH A graph whose vertices are exonic segments and edges are the connection among the vertices. Each vertex has a set of incoming and outgoing edges.

COMPONENT A subgraph of the splicing graph. All components are disconnected. Every vertex/edge belongs to a unique component. There is no edge between any vertices from different components.

1

Introduction

1.1 Transcriptomics

The sequencing of the human genome in 2001 is a milestone in the scientific landscape and a springboard for genetic studies (1). With the availability of the whole human genome (GRCh37/hg19), researchers easily identify disease-causing mutations in more than 2850 genes that are responsible for a large number of Mendelian disorders. They also detect statistically significant associations of about 1100 loci to more than 165 complex diseases and traits (2).

Nonetheless, studying human genetic disorders is a complex task, especially for multifactorial diseases like cancer and neurodegenerative diseases (ND) (3). Through genome-wide association studies (GWAS), about 88% of the genetic variants (single nucleotide polymorphisms (SNPs)) associated to complex diseases and traits are found to be located within intronic or intergenic regions (4). This evidence strongly indicates that these mutations are likely to have causal effects by influencing gene expression rather than affecting protein function. Thus, despite a deep genetic knowledge for many human genetic diseases, to date most of the studies do not provide relevant clues about the real contribution, or the functional role, of such DNA variations to disease onset.

In this scenario, whole-transcriptome analysis (termed *transcriptomics* (5)) is increasingly acquiring a pivotal role as it represents a powerful discovery tool for giving functional sense to the current genetic knowledge of many diseases.

The transcriptome is the complete set of transcripts in a cell, and their quantity, for a specific developmental stage or physiological condition. It is indicative of gene activity. Identifying the full set of transcripts, including large and small RNAs, novel transcripts from unannotated genes, splicing isoforms and gene-fusion transcripts serves as the foundation for a comprehensive study of the transcriptome (6). The key aims of transcriptomics are: to catalogue all species of transcripts, including mRNAs, non-coding RNAs and small RNAs; to determine the transcriptional structure of genes, in terms of their start sites, 5' and 3' ends, splicing patterns and other post-transcriptional modifications; and to quantify the changing expression levels of each transcript during development and under different conditions (7).

A transcriptome consists of a small percentage of the genetic code that is transcribed into RNA molecules - estimated to be less than 5% of the genome in humans (8). By studying transcriptomes, we hope to determine when and where genes are turned on or off in various types of cells and tissues. The number of transcripts can be quantified to get some idea about the level of gene activity or expression in a cell.

Besides GWAS studies, transcriptome analysis is a very powerful tool for various applications. The transcriptome of stem cells and cancer cells is of particular interest for researchers who seek to understand the processes of cellular differentiation and carcinogenesis (9). And the transcriptome of human oocytes and embryos is utilized to understand the molecular mechanisms and signaling pathways controlling early embryonic development. It could theoretically be a powerful tool in making proper embryo selection in *in vitro* fertilisation (10).

1.2 Complex Transcriptome

Over the past decade, advances in high throughput sequencing and innovations in biochemical techniques have revealed a complex picture of the eukaryotic transcriptome (7).

A gene can be expressed to different proteins with diverse biological functions. The key regulation mechanism is named *alternative splicing*, which keeps only a set of selected exons during transcription. Different combinations of exons result in proteins with different functions. Considering that only 1.2% of the transcribed

RNAs are finally translated to produce proteins (8), the regulated process alternative splicing is playing a key role during gene expression. In this process, particular exons of a gene may be included within, or excluded from, the final processed messenger RNA (mRNA), resulting differences in the proteins from alternatively spliced mRNAs. Notably, alternative splicing allows the human genome to direct the synthesis of many more proteins than would be expected from its 20,000 protein-coding genes.

Alternative splicing is essentially universal in human multi-exon genes. Most genes that contain three or more exons give rise to alternative isoforms that may vary with the cell types or states. And these alternative spliced forms often have different, even antagonistic functions (11). For example, Figure 2.3 illustrates the spliced variants of human gene LRRCC1. In human genome, more than 75% of the genes have at least three exons (12) (Figure 1.1).

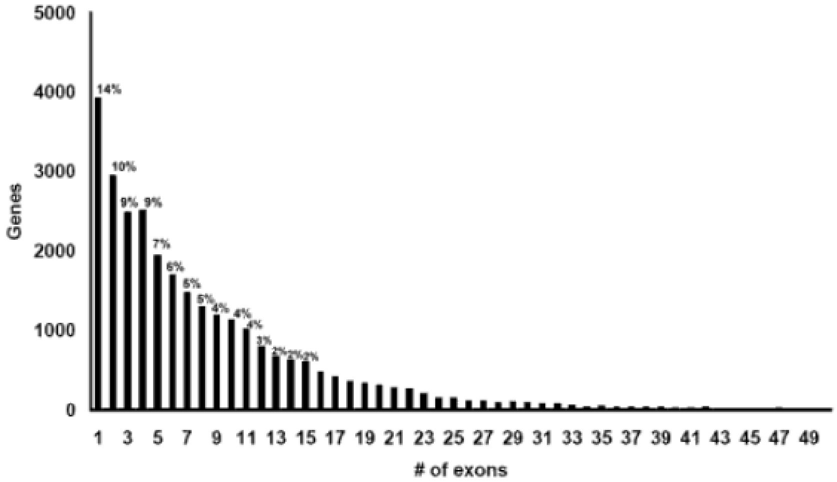


Figure 1.1: Distribution of number of genes against number of exons - Only 24% of the genes contain less than three exons.

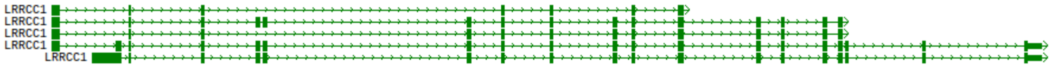


Figure 1.2: Transcript variants of gene LRRCC1 - All 5 transcript variants of gene LRRCC1 annotated in UCSC.

Based on our observations, out of the 22,680 protein-coding genes annotated in Ensembl database, 81.6% of them have at least two transcript variants. The distribution is shown in Figure 1.3.

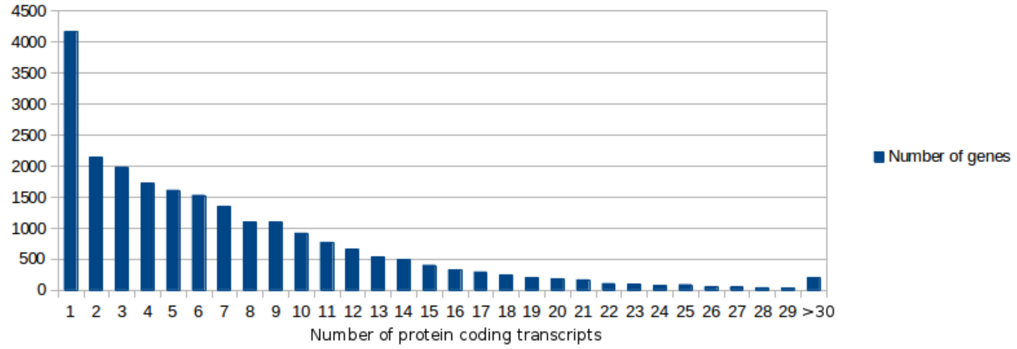


Figure 1.3: Distribution of number of protein-coding genes against number of transcript variants - There are totally 4,164 genes with only one transcript variant.

In an extreme case, the Drosophila Dscam gene generates more than 1,000 isforms, which are hypothesized to provide distinct identities to individual neuronal dendrites and to avoid self-interaction between the processes of a single neuron (13).

Moreover, long intergenic noncoding RNAs (ncRNAs) have been discovered more than the protein coding RNAs, exceeding 23,000 transcriptional units in mouse (14, 15). Many genes utilizes multiple promoters, and the position of the RNA 5' transcription start sites may shift under different environmental conditions.

1.3 Transcriptome Analysis and Gene Expression

Sequencing of RNA has long been recognized as an efficient method for gene discovery and remains the gold standard for annotation of both coding and noncoding genes (16). There are mainly two categories of technologies to deduce and quantify the transcriptome, i.e., hybridization-based and sequencing-based approaches. Hybridization-based approaches typically involve incubating fluorescently labelled cDNA with custom-made microarrays or commercial high-density oligo microarrays (17, 18, 19). Specialized microarrays have also been designed. For example, arrays with probes spanning exon junctions can be used to detect and quantify distinct splicing isoforms (20). Hybridization approaches have high throughput and relatively low cost. But they rely upon existing knowledge about the genomic sequences. They also require high background levels owing to cross-hybridization (21). Moreover, comparing expression levels across different experiments is often difficult and can require complicated normalization methods.

Sequence-base approaches directly determine the cDNA sequences by traditional Sanger sequencing technology. Initially, cDNA or Expressed Sequence Tag (EST) libraries are sequenced (22, 23). But it suffers from low throughput, expensive cost and generally not quantitative. Another set of tag-based methods are then developed to overcome these limitations. They include serial analysis of gene expression (SAGE) (24, 25), cap analysis of gene expression (CAGE) (26), and massively parallel signature sequencing (MPSS) (27). Tag-based approaches give high throughput and high resolution gene expression analysis. But the clear shortcoming is that they are based on expensive Sanger sequencing. Moreover, only some of the transcripts are analysed and isoforms are generally not distinguishable from each other.

Recently, advances in RNA sequencing are achieved as a result of new sequencing methods called Next Generation Sequencing (NGS), which generates large volume of short reads, providing high resolution to single nucleotide base. The details are included in next section.

1.4 Next Generation Sequencing

Maxam-Gilbert sequencing and Sanger sequencing (28) are called first generation sequencing technologies. Although they are introduced at the same time, Sanger sequencing becomes the golden standard due to its higher efficiency and lower radioactivity. The sequencing cost and speed are improved continuously. The human genome project uses Sanger sequencing to construct the euchromatic sequence of the human genome (29). In 2005, the 454 sequencer publishes a significant improvement in sequencing technologies. It sequences the genome of *Mycoplasma genitalium* in a single run (30). In 2008, the 454 sequences the genome of James Watson (31), marking another milestone in the extraordinarily fastmoving sequencing field. The advantages in throughput, cost and speed brought forward by 454 are remarkable. It marks the beginning of the Next Generation Sequencing (NGS) technologies, also known as the Second Generation Sequencing (SGS) technologies.

Competitors appear within a short time. In 2006, scientists from Cambridge introduce the Solexa 1G sequencer, claiming to resequence a human genome for about $100,000 within three months (32). In the same year, another competing sequencer the Agencourts SOLiD comes to the commercial market. It is also able to sequence complex human genome with comparable cost and speed. All of the three companies are acquired by more established companies (454 by Roche, Solexa by Illumina and Agencourt by ABI). More commercial sequencers are also provided by the Polonator (Dover/Harvard), the HeliScope Single Molecule Sequencer technology (Applied Biosystems and Helicos) and PacBio (Pacific Biosciences).

Comparing with traditional Sanger sequencing, NGS techniques are based on cyclic-array (33). Different sequencing platforms are quite diverse in sequencing biochemistry as well as in how the array is generated, but the work flows are conceptually similar (34). In shotgun sequencing with cyclic-array methods, common adaptors are ligated to the fragmented genomic DNA, which is then subjected to different protocols that give an array of millions of spatially immobilized PCR colonies or *polonies*. Then the polonies are tethered to a planar array, after which a single microliter-scale reagent volume is applied to manipulate the arrays in a highly paralleled manner. Finally imaging-based detection is used to acquire sequences on all tethers in parallel.

NGS platforms provide sequencing services with higher throughput and much lower cost. Figure 1.4 and 1.5 show the dramatical drop of the sequencing costs per genome and per Mb (35) since 2001.

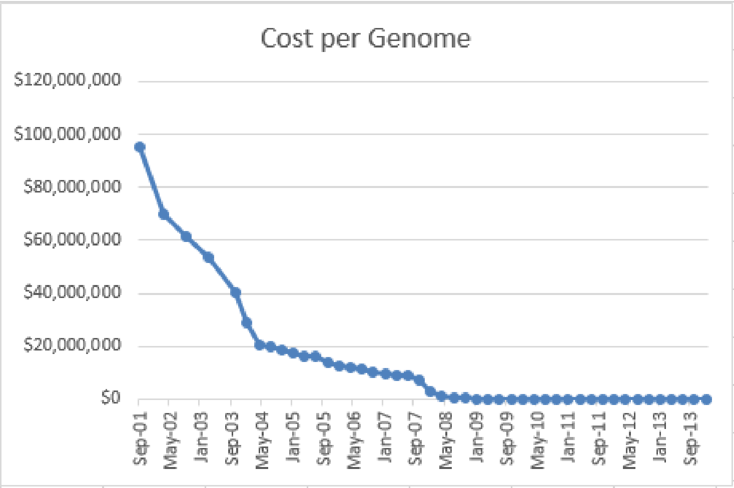


Figure 1.4: Cost per genome - The sequencing cost per genome from Sep 2001 to Jan 2014. Source: http://www.genome.gov/sequencingcosts/

NGS motivates a vast volumn of applications, allowing for huge advances in many fields related to the biological sciences (36). Figure 1.6 briefs some of the important NGS applications in the academy and industry (37).

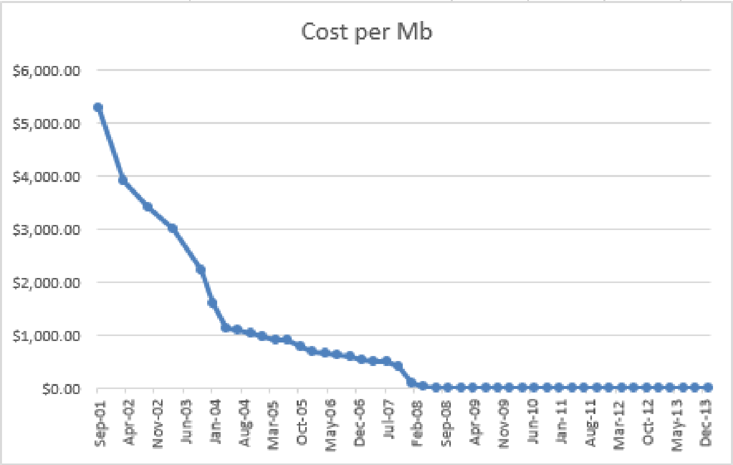
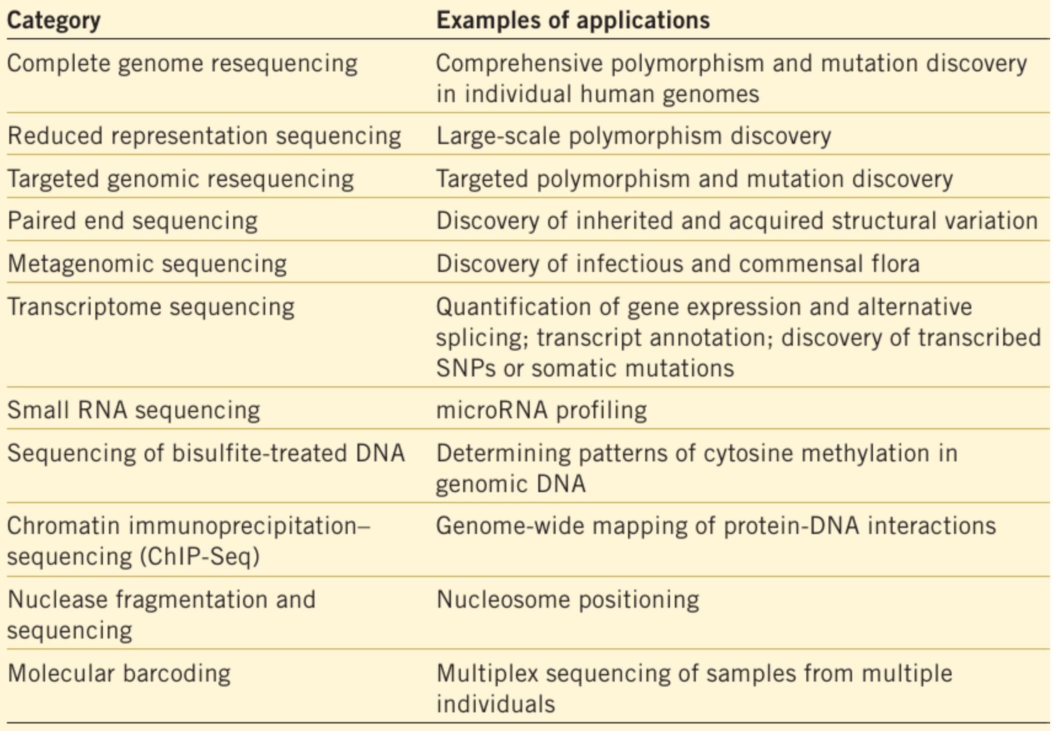


Figure 1.5: Cost per Mb - The sequencing cost per Mb from Sep 2001 to Jan 2014. Source: http://www.genome.gov/sequencingcosts/



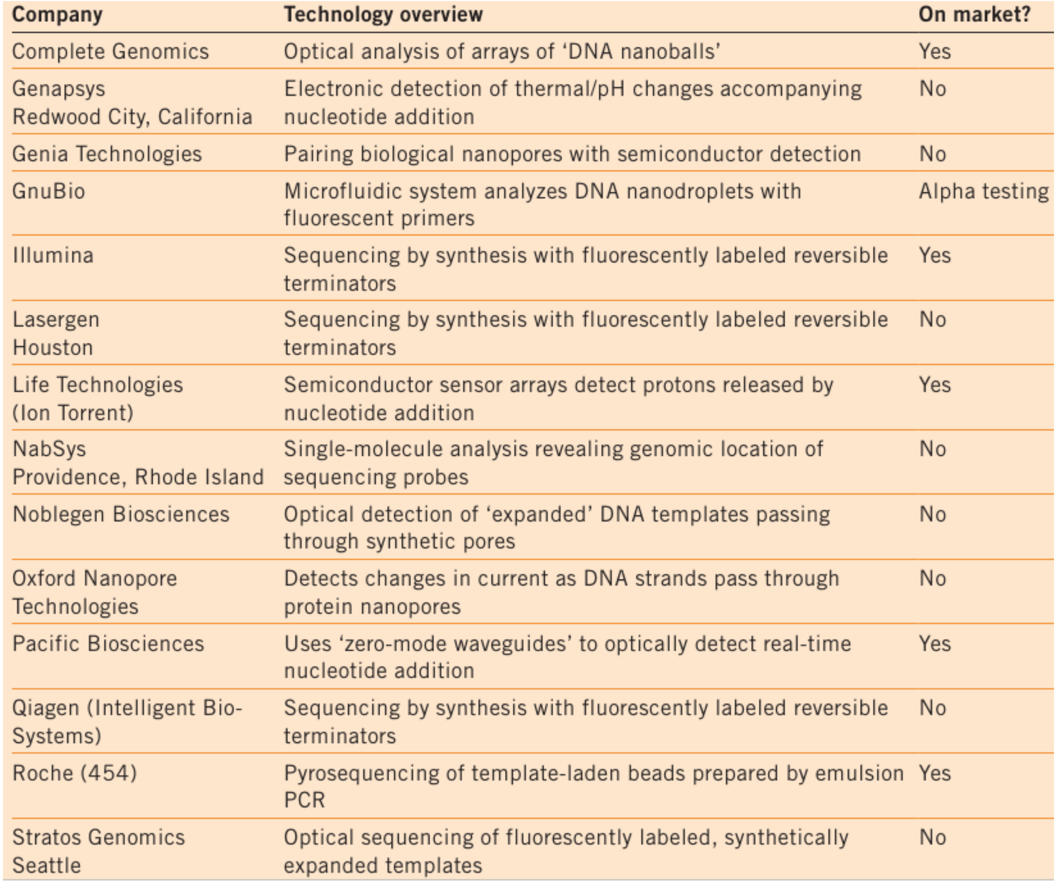
Category	Examples of applications
Complete genome resequencing	Comprehensive polymorphism and mutation discovery in individual human genomes
Reduced representation sequencing	Large-scale polymorphism discovery
Targeted genomic resequencing	Targeted polymorphism and mutation discovery
Paired end sequencing	Discovery of inherited and acquired structural variation
Metagenomic sequencing	Discovery of infectious and commensal flora
Transcriptome sequencing	Quantification of gene expression and alternative splicing; transcript annotation; discovery of transcribed SNPs or somatic mutations
Small RNA sequencing	microRNA profiling
Sequencing of bisulfite-treated DNA	Determining patterns of cytosine methylation in genomic DNA
Chromatin immunoprecipitation–sequencing (ChIP-Seq)	Genome-wide mapping of protein-DNA interactions
Nuclease fragmentation and sequencing	Nucleosome positioning
Molecular barcoding	Multiplex sequencing of samples from multiple individuals

Figure 1.6: NGS applications - The applications accelerated by NGS technologies

In the following subsections, we check existing NGS platforms and then brief three major NGS applications.

1.4.1 NGS Platforms

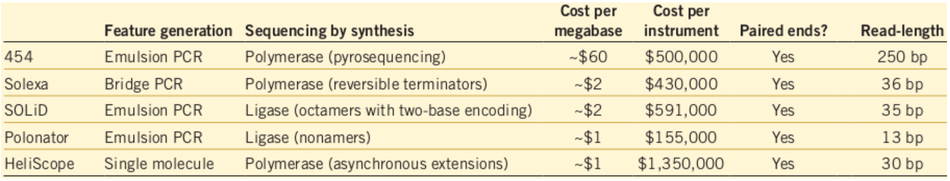
As costs fall and sequencing quality climbs, NGS sequencers are no longer confined to a handful of high-powered genomics centers, but are appearing in even small laboratories (38). A substantial proportion of researchers carry out their NGS activities at commercial service provider. Figure 1.7 is a complete list of current NGS platforms in academy and industry (38). Based on some marketing surveys (39), Illumina HiSeq 2000/1000 is the most popular NGS platform in the market (more than 30% of the respondents).



Company	Technology overview	On market?
Complete Genomics	Optical analysis of arrays of 'DNA nanoballs'	Yes
Genapsys Redwood City, California	Electronic detection of thermal/pH changes accompanying nucleotide addition	No
Genia Technologies	Pairing biological nanopores with semiconductor detection	No
GnuBio	Microfluidic system analyzes DNA nanodroplets with fluorescent primers	Alpha testing
Illumina	Sequencing by synthesis with fluorescently labeled reversible terminators	Yes
Lasergen Houston	Sequencing by synthesis with fluorescently labeled reversible terminators	No
Life Technologies (Ion Torrent)	Semiconductor sensor arrays detect protons released by nucleotide addition	Yes
NabSys Providence, Rhode Island	Single-molecule analysis revealing genomic location of sequencing probes	No
Noblegen Biosciences	Optical detection of 'expanded' DNA templates passing through synthetic pores	No
Oxford Nanopore Technologies	Detects changes in current as DNA strands pass through protein nanopores	No
Pacific Biosciences	Uses 'zero-mode waveguides' to optically detect real-time nucleotide addition	Yes
Qiagen (Intelligent Bio-Systems)	Sequencing by synthesis with fluorescently labeled reversible terminators	No
Roche (454)	Pyrosequencing of template-laden beads prepared by emulsion PCR	Yes
Stratos Genomics Seattle	Optical sequencing of fluorescently labeled, synthetically expanded templates	No

Figure 1.7: NGS platforms - Existing NGS sequencers. Some of them are termed *Third Generation Sequencing*, such as PacBio

Figure 1.8 lists the cost of mainstream sequencers in 2008 (34). Since the initiation of 1000 genome project, the cost of sequencing an individual genome has been rapidly decreasing and will likely reach $1000 per person within in near future (37).



	Feature generation	Sequencing by synthesis	Cost per megabase	Cost per instrument	Paired ends?	Read-length
454	Emulsion PCR	Polymerase (pyrosequencing)	~$60	$500,000	Yes	250 bp
Solexa	Bridge PCR	Polymerase (reversible terminators)	~$2	$430,000	Yes	36 bp
SOLiD	Emulsion PCR	Ligase (octamers with two-base encoding)	~$2	$591,000	Yes	35 bp
Polonator	Emulsion PCR	Ligase (nonamers)	~$1	$155,000	Yes	13 bp
HeliScope	Single molecule	Polymerase (asynchronous extensions)	~$1	$1,350,000	Yes	30 bp

Figure 1.8: Cost of NGS platforms - The cost is based on survey in 2008.

1.4.2 Whole Genome Sequencing and GWAS

Emergence of NGS techniques boosts a huge wave of whole genome sequencing. According to the online GOLD database of Complete Genome Projects, there are totally 18,940 genomes sequenced up to now. Majority of them are finished within the last 20 years. More and more species, such as Baiji (*Lipotes vexillifer*) (40) and mulberry tree Morus notabilis (41), are being sequenced. The existence of reference genome largely aids the understanding of all related fields.

Sequence analysis has been widely used to guide the therapy of various complex diseases such as cancer (42, 43). The NGS approach holds advantages over traditional methods, including the ability to fully sequence large numbers of genes in a single test and simultaneously detect deletions, insertions, copy number alterations, translocations, and exome-wide base substitutions in all known cancer-related genes. It is much easier and cheaper to sequence the whole genome of patients at different stages, such that studying the development of the cells is possible.

All of these initiate the substantial advances in Genome-Wide Association Study (GWAS). A genome-wide association study is an approach that involves rapidly scanning markers across the partial or complete set of genomes, of many people to find genetic variations associated with a particular disease (44). With the association information, researchers are able to develop better strategies to diagnose, treat and prevent the diseases. The advances include type 1 (45) and type 2 diabetes (46), inflammatory bowel disease (47), etc.

GWAS typically focuses on the associations between Single Nucleotide Polymorphism (SNPs) and traits of complex diseases. The associated SNPs are then considered to mark a region of the human genome which influences the existence of diseases. Researchers usually sequence the genome of tumor and normal samples to identify the associations. More and more studies utilize NGS sequencing to obtain transcriptome of the samples and analyze the different sets of SNPs and genes expressed (48, 49, 50).

1.4.3 ChIP-Seq

ChIP-Seq combines chromatin immunoprecipitation (ChIP) with massively parallel DNA sequencing to identify binding sites of DNA-associated proteins (51). The main purpose of ChIP-seq is to generate a genome-wide map of a variety of histone modifications to define different types of chromatin domains and their relationship to the regulatory state of genes.

In 2007, the Solexa massively parallel sequencing technique is applied to chromatin-immunoprecipitated material from human CD4+ T cells (52), where two DNA-binding proteins - RNA polymerase II (RNA POL II) and the chromatin boundary marker CTCF - are analyzed. In addition, the ENCODE and modENCODE consortia have designed and performed more than a thousand individual ChIP-seq experiments for more than 140 different factors and histone modifications in more than 100 types of cells from four different organisms D. melanogaster, C. elegans, mouse, and human (53).

1.4.4 RNA Sequencing

RNA-seq is a technology that uses the capabilities of NGS techniques to reveal a snapshot of RNA presence and quantity of a particular cell at a given moment, restricted in some circumstances. Since our first study is utilizing RNA-seq data, we are going to discuss more details about the advantages, data characteristics and bioinformatics applications of RNA-seq in next chapter.

1.5 Challenges of NGS

We have described various advantages brought by Next Generation Sequencing. However, NGS introduces more computational and management challenges due to higher error rates, shorter read length and unprecedented volumes of data. Table 1.1

compares the data characteristics between NGS sequencers and traditional Sanger sequencing techniques (54).

Sequencer	**454 GS FLX**	**HiSeq 2000**	**SOLiDv4**	**Sanger 3730Xl**
Sequencing mechanism	Pyrosequencing	Sequencing by synthesis	Ligation, two-base coding	Dideoxy chain termination
Read length	700bp	50SE, 50PE, 101PE	50+35bp, 50+50bp	400-900bp
Accuracy	99.9%	98%	99.94%	99.999%
Reads	1M	3G	1200-1400M	-
Output data/run	0.7G	600G	120G	1.9-84Kb
Time/run	24 Hours	3-10 days	7 days for SE 14 days for PE	20 Mins - 3 Hours

Table 1.1: Comparison of data characteristics

Various bioinformatics tools are developed to capture the NGS wave (34). Some important computational tools include: (i) full/spliced alignment of short reads to reference genome; (ii) base-calling and/or polymorphism detection; (iii) genome/transcriptome assembly from single-end or paired-end reads; (iv) genome annotation, management and visualization.

The most demanding challenge is the overwhelming NGS data. Considering the capabilities of current computers, the data processing time falls far behind the data generation rates. This doesn't even count the time to perform thorough data analysis. High performance computing and cloud computing are steadily applied to the NGS data processing and management.

According to a recent survey conducted by Bio IT World (39), more than 50% of the 232 respondents suggest that the biggest challenge for NGS to move to the clinic is data analytics and data management. For example, we have observed that for an identical SNP, there exists multiple identities in public datasets. That results in ambiguities and confusion for researchers. In our second study UASIS, we actually propose an integrated platform for better SNP management.

1.6 Contributions of the Thesis

With the rapidly evolving NGS technologies, overwhelming NGS data has posed critical challenges to the whole bioinformatics community. In this thesis we introduce two powerful tools, PETA and UASIS, for better interpretation and management of the NGS data.

We have developed a *de novo* transcriptome assembly tool PETA (Paired End Transcriptome Assembler) to efficiently construct accurate and full-length transcripts from RNA-seq reads, without the existence of the reference genome.

Although researchers have sequenced a large number of genomes in the last 20 years, a lot of studies are conducted without the reference genome. Due to complexity of eukaryotic species, they are difficult to sequence completely. According to the statistics from GOLD Genomics Online Database (55), currently the number of completed eukaryotic genomes is 918, which is much lower than the number of bacterial genomes (17,692).

Our assembler contributes to the transcriptomics study by providing a powerful tool to reconstruct a full picture of transcriptome in the cell. PETA, as the name indicates, is tailored for paired-end RNA-seq reads. PETA is based on a classical overlap-layout-consensus strategy to grow longer contigs. The reads supported by their mates will be weighted heavily to contribute more to the determination of next base. It also ensures that every transcript reported is supported by paired-end reads whose insert size is within the correct range. We utilize the full-length paired-end reads to construct a simpler, cleaner and more reliable graph structure and capture all splicing patterns in a conservative manner.

The experiments on *Schizosaccharomyces pombe* and human RNA-seq datasets shows advanced features comparing with existing assemblers.

In the second study UASIS (Universal Automated SNP Identification System), we propose a novel SNP nomenclature, which use unique information of a SNP to define the identities. The universal nomenclature is informative, unambiguous and easy to maintain.

Meanwhile, we develop three utilities, namely UASIS Aligner, Universal SNP Name Generator and SNP Name Mapper. The integrated application maps the SNP identities from different databases, including dbSNP, GWAS, HapMap and

JSNP etc. It is extremely useful when the researchers are working on literature of specific SNPs.

1.7 Organization of the Thesis

Here is the organization of the remaining content of the thesis. In Chapter 2, we brief some biological backgrounds to help understand the thesis better. We also introduce more details about RNA-seq protocols and applications. Chapter 3 discusses the problem of transcriptome assembly and existing approaches. The Problem Statement can be found in Chapter 4, where we formulate the transcriptome assembly problem in a systematic manner. Meanwhile, we illustrates the workflow of PETA in a global view. Chapter 5 focuses on the hashing strategies we utilize for fast pairwise alignment, which is needed to pick overlapping reads efficiently. Chapter 6 and Chapter 7 describe the core implementation of our assembler, including read extension, graph construction and transcripts extraction. In Chapter 8 we show and analyze the experimental results on two real RNA-seq datasets.

In Chapter 9, we introduce the novel integrated system UASIS in the perspective of data management. We discuss problems of current SNP nomenclatures and then introduce the implementations of UASIS. Finally we conclude the thesis.

2

Basic Biology and RNA Sequencing

In this chapter, we brief some biological backgrounds, including Single Nucleotide Polymorphism (SNP) and RNA-seq protocol. We summarize the RNA-seq techniques and the data characteristics. Since PETA is tailored to paired-end RNA-seq reads, we will introduce the paired-end protocols. The emerging/future RNA-seq techniques are also introduced.

2.1 Basic Biology

2.1.1 DNA

Human beings are keeping high enthusiasm in understanding the nature. How does the life evolves? Why are some people healthy and others ill? In April 1953, James Watson and Francis Crick present the double helix structures of Deoxyribonucleic acid, or DNA, starting another amazing era. The sentence *"This structure has novel features which are of considerable biological interest"* may be one of science's most famous statements (56).

DNA is the molecule that carries genetic information from one generation to the other. Almost all species - bacteria, plants, yeast and animals - use DNA as the same building blocks, except that some viruses use RNA instead. Most DNA molecules consist of two biopolymer strands coiled around each other to form a double helix structure. The two DNA strands are composed of four kinds of nitrogen-containing nucleotides: guanine (G), adenine (A), thymine (T), and cytosine (C), as well as a monosaccharide sugar called deoxyribose and a phosphate group. The nucleotides are paired following the base pairing rules (A with T and C with G). Hydrogen bonds bind the nitrogenous bases of the two separate polynucleotide strands to make double-stranded DNA. Figure 2.1 illustrates the DNA structure.

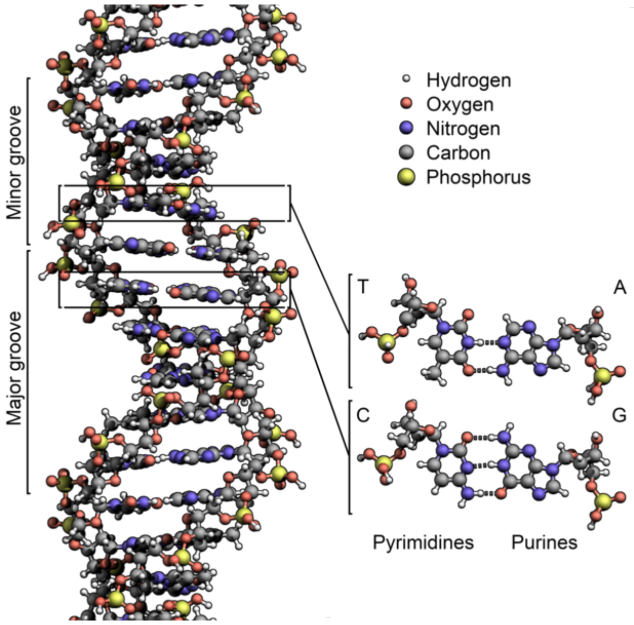


Figure 2.1: Double helix structure of DNA - DNA is a winning formula for packaging genetic material. The structure is identical within almost all species.

DNA strands have directionality. One end of a DNA polymer contains an exposed hydroxyl group on the deoxyribose; this is known as the 3' end of the molecule. The other end contains an exposed phosphate group; this is the 5' end. In conversion, we also name the direction of the strand from 5' to 3' as the *forward* direction, and the opposite direction is named the *backward* direction.

Usually, we do not take the 3-dimensional structure into consideration. Instead, we use only sequential nucleotide bases to represent the DNA. For example, the human genome is composed of approximately 3 billion base pairs. However, the real topology of DNA is more complex. The two strands may be bend to interact with specific proteins during gene expression.

2.1.2 Single Nucleotide Polymorphism (SNP)

SNP, or Single Nucleotide Polymorphism, is defined as a polymorphism at a single base with a frequency of more than 1% in the population (57, 58). Alternative bases at the locus of SNPs are called *alleles*. They occur more frequently in non-coding regions than coding regions. On human genome, there is one SNP in every 300 nucleotides on average. Majority of the SNPs do not have affects on health. But some of them are proved to influence complex diseases. For example, the APOE gene influences postmenopausal osteoporosis through SNP-SNP interactions (59).

SNPs are the most common type of genetic variations among people. Around 90% of the genome variations are limited to SNPs (60). As of 13 May 2014, dbSNP has already collected 62,387,846 SNPs on human genome. They have been used in Genome-Wide Association Studies (GWAS), for instance, as high-resolution markers in gene mapping related to diseases or normal traits.

2.1.3 Gene

The concept of *gene* has evolved and becomes more complex (61). Generally speaking, "A gene is a locatable region of genomic sequence, corresponding to a unit of inheritance, which is associated with regulatory regions, transcribed regions, and or other functional sequence regions" (61, 62). It is a blueprint for a protein, which determines the functionality of the cells.

A gene consists of transcribed regions and regulatory regions. A typical structure of a gene is shown in Figure 2.2, where the exons will be transcribed to form RNA molecules and the introns will be spliced out. However, the same gene may be expressed differently in different cells, which means, a gene may produce different proteins depending on the regulations. In this case, the concept of an exon/intron is not absolute. As novel transcripts are keeping being discovered, some introns are found to be transcribed. In convention, as long as a DNA segment is transcribed into at least one RNA molecules, we categorize it to be an exon.

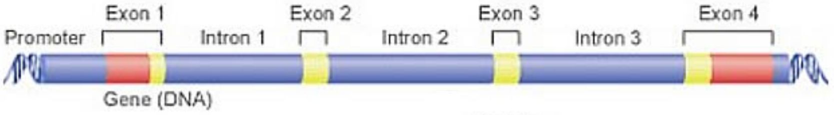


Figure 2.2: Gene structure - Exons and introns of the gene.

Despite of the importance of genes, only 1.5 percent of the DNA in the genome actually codes for genes (29). Majority portion of the genome is transcribed to introns, retrotransposons and seemingly a large array of noncoding RNAs (63, 64). The vast majority of the genome is far from well understood.

2.1.4 RNA and Alternative Splicing

Ribonucleic acid (RNA) is a family of large biological molecules that play important roles during gene expression. Cellular organisms use messenger RNA (mRNA) to convey genetic information using the nucleotides guanine (G), adenine (A), uracil (U), and cytosine (C). mRNAs direct synthesis of specific proteins, while many viruses encode their genetic information using an RNA genome.

There are also non-coding RNAs (ncRNAs) that are important in gene regulation. The most prominent ones are transfer RNA (tRNA) and ribosomal RNA (rRNA). A tRNA is a small RNA with about 80 nucleotides. It transfers a specific amino acid to a growing polypeptide chain at the ribosomal site of protein synthesis during translation. rRNAs are the catalytic component of the ribosomes. Other members of the large RNA family include mircoRNA (miRNA), piwi-interacting RNA (piRNA), small interfering RNA (siRNA) and many more.

Synthesis of a single strand RNA is usually catalyzed by the enzyme RNA polymerase using DNA as a template, a process known as *transcription*. The immature pre-mRNAs are often modified by enzymes after transcription. For example, *alternative splicing* removes the introns on the pre-mRNAs. Then another process *translation* will synthesize a protein using the mRNA as the template.

There are millions of proteins in human cells, while the number of protein-coding genes are approximated to be around only 20,000. Alternative splicing makes it possible for a gene to code for multiple different proteins. In this process, particular exons of a gene may be included within, or excluded from the processed mRNA. The process is illustrated in Figure 2.3. Alternative splicing is a normal phenomenon in eukaryotes. Based on our observations, more than 80% of the genes in Ensembl database record at least two transcript variants.

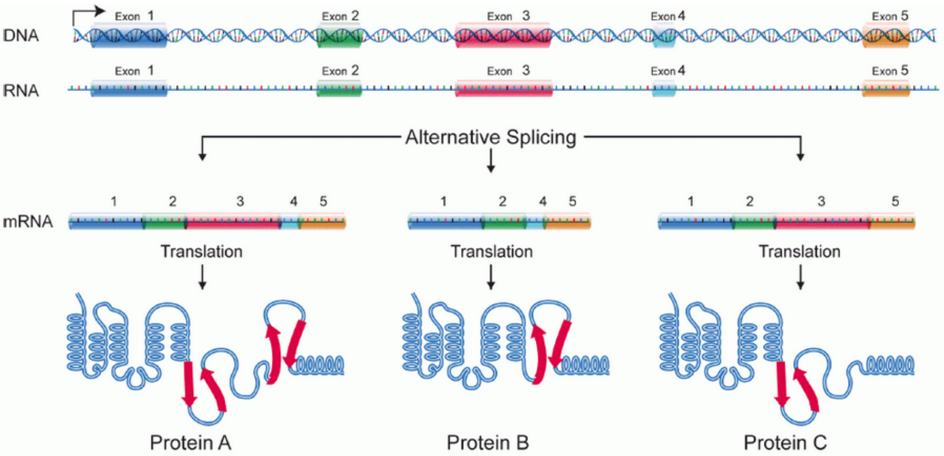


Figure 2.3: Transcript and translation - The same gene can be translated into three different proteins through alternative splicing.

2.1.5 Complementary DNA (cDNA)

In genetics, complementary DNA (cDNA) is the DNA sequence synthesized from a mRNA template in a reaction performed by the enzymes reverse transcriptase and DNA polymerase. cDNA is a synthesized chemical product, rather than a real molecule in the cells. Due to the single-strand feature and degradation, RNAs are more susceptible than DNA. In this case, the term *cDNA* is typically used to refer to an mRNA transcript's sequence, expressed as DNA bases (GCAT) rather than RNA bases (GCAU).

Complementary DNA is often used in gene cloning or as gene probes or in the creation of a cDNA library. To sequence a RNA, researchers usually synthesize the cDNA library at the first place.

2.1.6 Sequencing

Sequencing is the process of determining the primary structure of a stretch of biological molecules (DNA, RNA, etc.). The result is a symbolic linear depiction known as a sequence which succinctly summarizes much of the atomic-level structure of the sequenced molecule. A sequence is represented by strings of nucleotide bases (A, C, U/T, and G). Due to the double helix structure of DNA, the length of a sequence is usually in the unit of *base pair*, or *bp*. For example, the complete human genome is sequenced in 2004, with around 3 billion base pairs.

2.2 RNA Sequencing

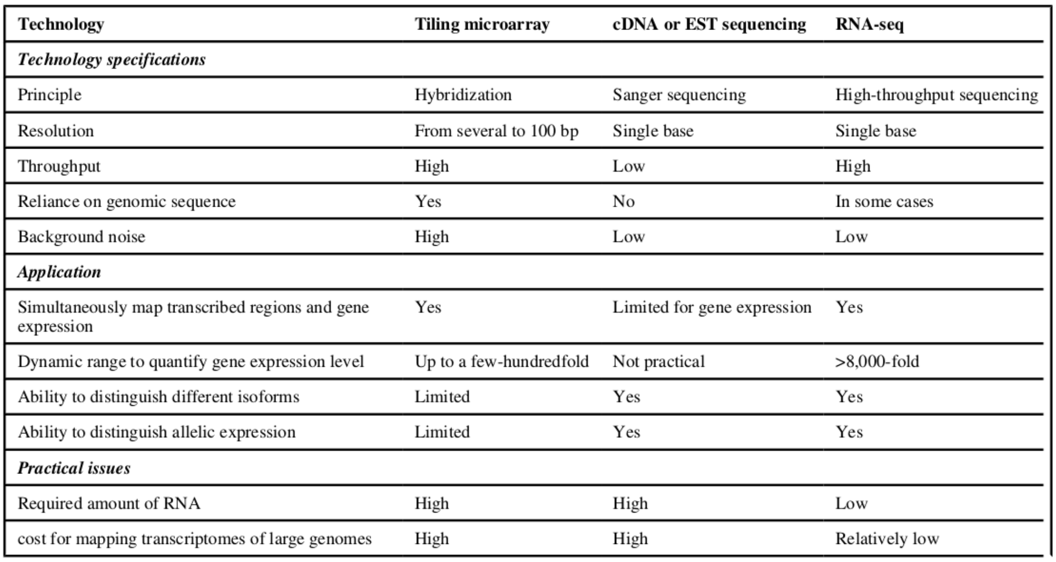
Before the complete of human genome in 2004, predictions about the protein-coding genes are error prone and the roles of noncoding RNAs (ncRNAs) are very limited. Introns, interspersed repeated sequences and transposable elements are considered as junk DNA and evolutionary debris, and alternative splicing is an exception rather than the rule.

In 2008, RNA-seq (RNA sequencing), which sequences the complete RNA collection using Next Generation Sequencing techniques at massive scale, starts to reveal the complex picture of various transcriptomes in a high resolution. It outperforms other techniques by providing lower cost, higher coverage, better resolution and faster speed. New methodologies of RNA-seq have been providing a progressively better understanding in the transcriptomes of prokaryotes and eukaryotes (65).

"RNA-seq is expected to revolutionize the manner in which eukaryotic transcriptomes are analyzed" (7). Since the first wave of RNA-seq applications introduced by (66, 67, 68, 69), RNA-seq has been applied to various transcriptome projects. All these studies bring more comprehensive understanding of transcription starting sites, the cataloguing of sense and anti-sense transcripts, improved detection of splicing patterns and fusion genes. It even allows the selection of specific RNA molecules before sequencing, allowing more focused studies on targeted molecules. Figure 2.4 compares three categories of RNA analysis techniques microarray, EST sequencing and RNA-seq (7).

Although diverse RNA-seq protocols use different approaches, all of them share a general idea as shown in Figure 2.5 (7, 65). First of all, a population of RNA (total or partial) is converted to a cDNA library with adaptors attached to one end or both ends. Each molecule, with or without amplification, is then deeply sequenced on some NGS platforms. Filtering strategies may be applied to clean and report the single-end or paired-end reads.

Meanwhile, advances of RNA-seq accelerate the developments of Genome-Wide Association Studies (GWAS), which help to diagnose, treat and prevent complex diseases such as diabetes and cancer (70).



Technology	Tiling microarray	cDNA or EST sequencing	RNA-seq
Technology specifications			
Principle	Hybridization	Sanger sequencing	High-throughput sequencing
Resolution	From several to 100 bp	Single base	Single base
Throughput	High	Low	High
Reliance on genomic sequence	Yes	No	In some cases
Background noise	High	Low	Low
Application			
Simultaneously map transcribed regions and gene expression	Yes	Limited for gene expression	Yes
Dynamic range to quantify gene expression level	Up to a few-hundredfold	Not practical	>8,000-fold
Ability to distinguish different isoforms	Limited	Yes	Yes
Ability to distinguish allelic expression	Limited	Yes	Yes
Practical issues			
Required amount of RNA	High	High	Low
cost for mapping transcriptomes of large genomes	High	High	Relatively low

Figure 2.4: Comparison of three RNA analysis techniques - RNA-seq provides single-base resolution, high coverage and reads with less noise.

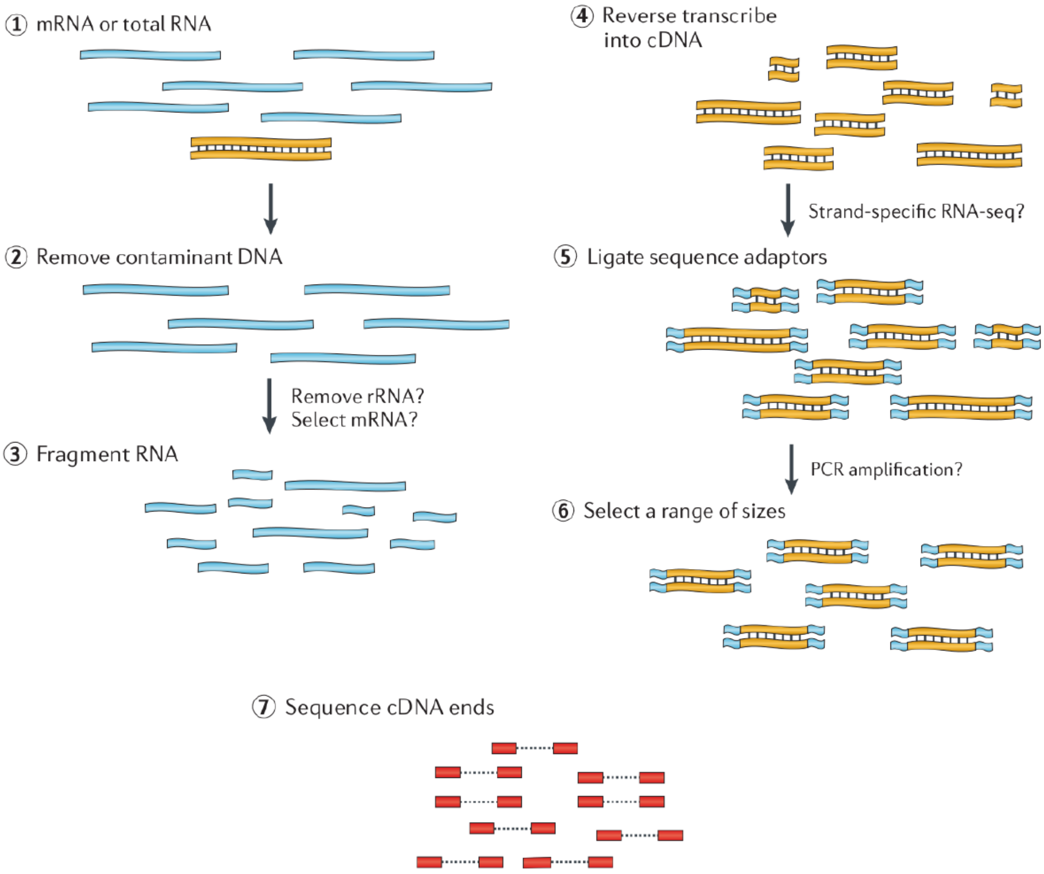


Figure 2.5: General Procedure of RNA-seq - The general process to generate RNA-seq reads.

2.3 Challenges of RNA-seq

Similar to other NGS techniques, RNA-Seq faces several computational challenges, including the development of efficient methods to store, retrieve and analyze large amounts of data. The bioinformatics tools must reduce errors in image analysis and base-calling and remove low-quality reads. We here discuss about the characteristics of RNA-seq data and the algorithmic challenges to develop supporting tools.

2.3.1 Sequencing Errors

All library construction approaches of RNA-seq experiments introduce unavoidable biases, which can lead to the erroneous interpretation of the data (71, 72).

The ideal approach should be able to identify and quantify all kinds of RNAs in full-length, including long mRNAs and other smaller regulation RNAs. During library construction, large RNA molecules must be fragmented into smaller pieces (200bp to 500bp) to be compatible with most deep sequencing technologies. The common methods for fragmentation include RNA fragmentation (RNA hydrolysis or rebulization) and cDNA fragementation (DNase I treatment or sonication). RNA fragmentation introduces little bias over the transcript body, while the transcript ends are depleted (7). Conversely, cDNA fragmentation favours the 3' end of the transcripts.

During the PCR amplification, it is known that not all fragments are amplified with the same efficiency. Many identical short reads can be obtained from the cDNA libraries. These could be genuine reflection of abundant RNAs, or may be PCR artefects. One way to distinguish these reads is to compare reads from multiple replicates.

Moreover, producing strand-specific RNA-seq data is currently laborious because of many extra tedious steps or direct RNA-RNA ligation (69).

Biases also happen for RNA-seq extraction using Trizol (73). Selective loss occurs for GC poor or highly structured small RNAs at low RNA concentrations. There are many more errors can be introduced during library preparation (71).

Sequencing errors occur in the RNA-seq data as a result of mistakes in base calling or the insertion/deletion of a base. For example, the error rate of Illumina GenomeAnalyzer is up to 3.8%. PacBio, which produces longer reads with length of around 2500bp, reports a error rate as high as 15%. Although error correction algorithms are developed (74), it is still a problem for RNA-seq applications.

2.3.2 RNA-seq Alignment

Once the short RNA-seq reads are obtained, the first task is to map the reads to the reference genome. There are powerful pairwise alignment tools, such as MAQ (75), Bowtie/Bowtie2 (76) and BWA (77). However, due to alternative splicing, some short transcriptomic reads span the exon junctions. Such that two portions of the reads should be aligned to two different positions on the genome. For complex transcriptomes it is even more difficult since alternative splicing occurs more frequently.

For large transcriptomes, alignment is complicated because a read can be uniquely mapped to multiple locations on the genome. Short reads from highly repetitive regions have high copy numbers. A possible solution is to assign the multi-matched reads based on the reads mapping to their neighbouring unique regions. Alternatively, if the RNA-seq is constructed following a paired-end protocol, which sequences both ends of a DNA fragment, the multi-matched reads can be assigned to a unique locus based on their paired reads.

A lot alignment tools are developed to map the spliced reads, including the BLAST-like alignment tool (Blat) (78), GEM (79), MapSplice (80) and TopHat (81).

2.3.3 Transcriptome Assembly

Transcriptome assembly is another important fundamental application for downstream analysis. It assembles contigs/transcripts which can be used to identify and quantify the genes expressed in the sample. Based on the assembly strategies, there are three kinds of assemblers. The transcripts are assembled with or without the reference genome. And some transcriptome assemblers combine the two strategies to achieve better results. We describe more details about this topic in next Chapter.

2.4 Paired-end RNA-seq

A RNA-seq library can be designed to be paired-end (PET), which provides extra information for transcriptomics. The principal concept of the PET strategy is the extraction of only short tag signature information from both ends of target DNA fragments. The distance between pairs of reads can be estimated based on sequencing protocol. By mapping the paired tag sequences to reference genomes, researchers are easier to determine the boundaries of the target DNA fragments in the genome landscape. The process is illustrated in Figure 2.6 (82).

Paired-end RNA-reads provide extra information to determine the origin of the reads. The distance between paired reads (or *insert size*) is roughly 200bp to 500bp, which is able to go across large portion of repetitive regions. For transcriptomics, the paired-end reads can be utilized to identify novel splicing events and fusion genes (83). Our assembler PETA makes full use of the paired-end information to reconstruct accurate transcripts.

2.5 Long Read RNA-seq

As NGS technologies evolve rapidly, read length from third generation RNA sequencers is getting longer. Pacific Biosciences (PacBio) develops a pioneering technique SMRT (84), short for *single molecules real-time*, to provide commercial *Long Read RNA-seq* service. The sequencer PacBio RS is capable of generating reads up to several kilobases (averaging 3,146 bases), which may cover a single transcript to its full length (85, 86) without any assembly process. In future, if this technology reaches a throughput that is comparable to the second-generation technologies, the transcriptome analysis would be much easier. The assembly process will be probably eliminated (6).

PacBio is capable of generating sequence without bias. It is also able to generate regions with high GC content. However, there are limitations to apply the long read RNA-seq to practical applications (86, 87). First of all, the error rate is too high to be acceptable. In experiments, the sequencing error rate is as high as 15%. Secondly, the throughput is moderate (50,000 reads per single molecule real time (SMRT) cell). Meanwhile, advantages in read length come at a much greater cost per nucleotide (87).

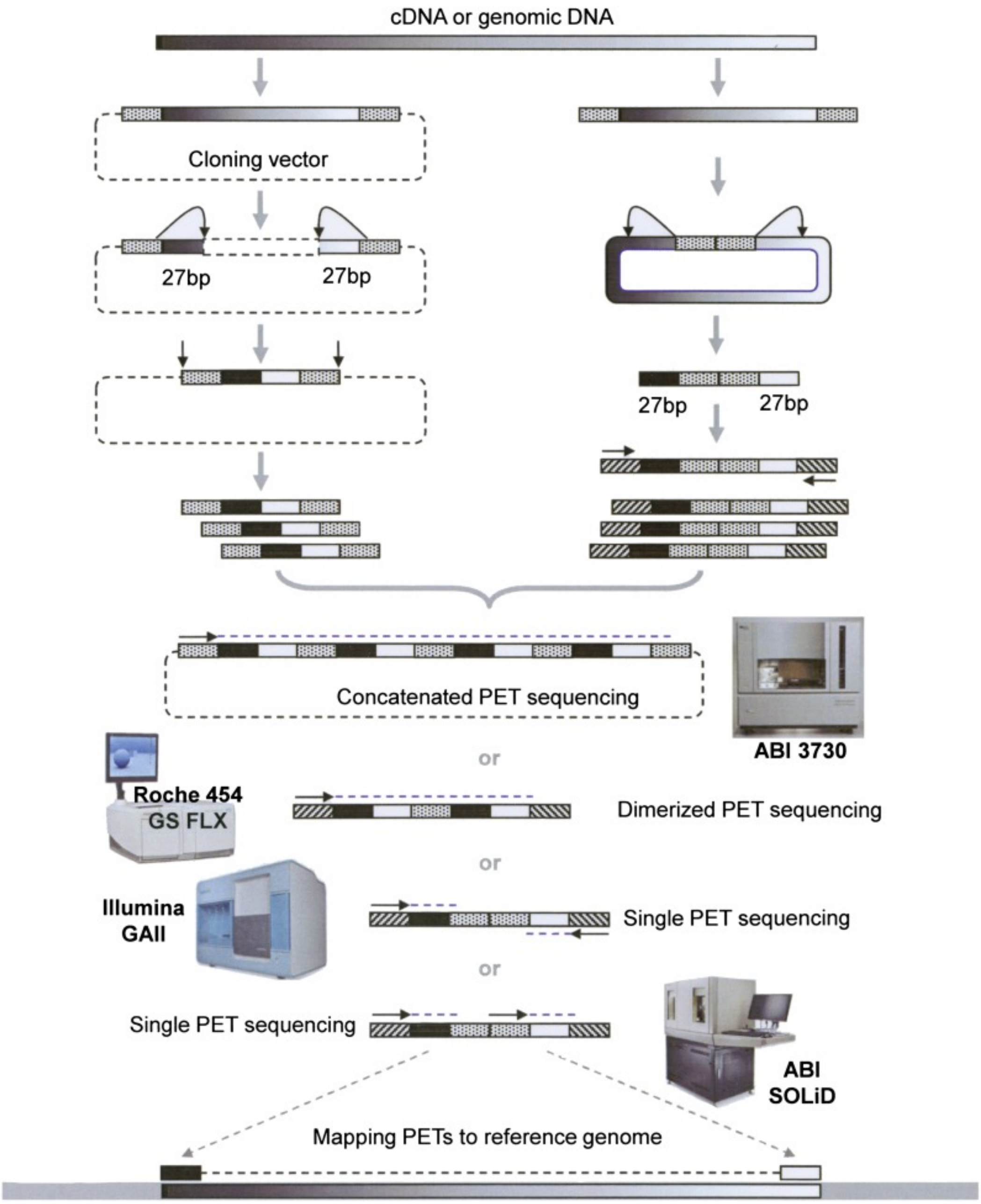


Figure 2.6: Schematic view of PET methodology - PET construction can be done through cloning-based or cloning-free procedures. Most NGS sequencers support paired-end sequencing.

PacBio provides error correction tools to clean the reads. Some researchers combine the reads from second and third generation sequencing to obtain a comprehensive characterization of the transcriptome of the human embryonic stem cell (86). From this perspective, transcriptome assembly process is still an essential step for thorough analysis.

3

Transcriptome Assembly

3.1 Introduction

Compared with traditional Sanger sequencing technique, NGS platforms achieve significantly lower production costs and higher throughput (34). However, the reads produced by NGS are much shorter than Sanger reads, currently 400-500 basepairs (bp) for 454, 50-200bp for Illumina and 100bp for SOLiD. Large volume of NGS short reads pose significant challenges for bioinformatics tools. At the early stage of commercial availability, a variety of software tools are tailored to process and analyse the data. They include: (i) alignment of sequence reads to a reference; (ii) base-calling and/or polymorphism detection; (iii) *de novo* assembly, from paired or unpaired reads; and (iv) genome browsing and annotation. Shendure and Ji (34) gave a list of NGS tools available.

In this study, we focus on the assembly applications only. First of all, it involves piecing together millions of low quality, short reads. Typical RNA-seq libraries are very large (tens to hundreds of gigabases), which require strong computational power and large memory. A dozen genome assemblers are developed for NGS data, including ALLPATHS (88), Velvet (89), ABySS (90) and PE-Assembler (91), etc. But these tools cannot be directly applied to RNA-seq libraries.

First of all, DNA sequencing depth is supposed to be uniform across the whole genome. But the coverage of RNA-seq can vary by a several orders of magnitude. Genome assemblers frequently make use of the uniform coverage to perform error detection/correction and distinguish repeat regions, such as Pebble and Rock

Band algorithms in Velvet (89). But these approaches cannot be transplanted to transcriptome assembly directly. Secondly, availability of strand specific RNA-seq protocols is more common (92). We need to take advantage of the strand information to improve the performance (93, 94). Finally, the same gene could be expressed differently, resulting in various combinations of the exons. This makes genome assemblers inadequate to resolve the ambiguities (6).

The aim of a transcriptome assembler is to reveal the transcription structure from millions of short reads. Ideally, it should be able to report a set of transcripts under the particular environment, as well as all splicing patterns accurately. The overall strategy is to investigate the overlapping between reads and reconstruct the transcripts.

Currently, transcriptome assemblers mainly fall into three categories: reference-based (or *ab initio*), *de novo* (without a reference) and the combined strategy. Two leading reference based packages are Cufflinks (9) and Scripture (95). Figure 3.1 illustrates the overall strategy of the reference based assemblers (6). Generally speaking, all reads are mapped to the reference genome using a *splice-aware mapper* (Tophat (96) for Cufflinks and Scripture). Available mappers include SpliceMap (97), MapSplice (98) and GSNAP (99), etc. Then a graph is built by clustering the short reads. Finally, individual isoforms are determined after traversing the graph.

De novo approaches assemble the transcripts directly from the RNA-seq libraries without the help of reference genome. They are mostly based on *de Bruijn* graph (100, 101, 102, 103, 104) as shown in Figure 3.3. Different tools implement various customizations on the graph. Details of the *de Bruijn* graph and comparison of current *de novo* transcriptome assemblers are given in Section 3.2.

Some researchers talked about combining the former two strategies to create a more comprehensive transcriptome (105). By combining the two approaches, one can take advantages of the high sensitivity of reference based assemblers and leverage the strong capability of novel transcript detection of *de novo* assemblers.

3.2 Current Approaches

In this Section, we review *de novo* transcriptome assembly approaches specifically. Since most of state-of-art *de novo* assemblers are based on *de Bruijn* graph, we first

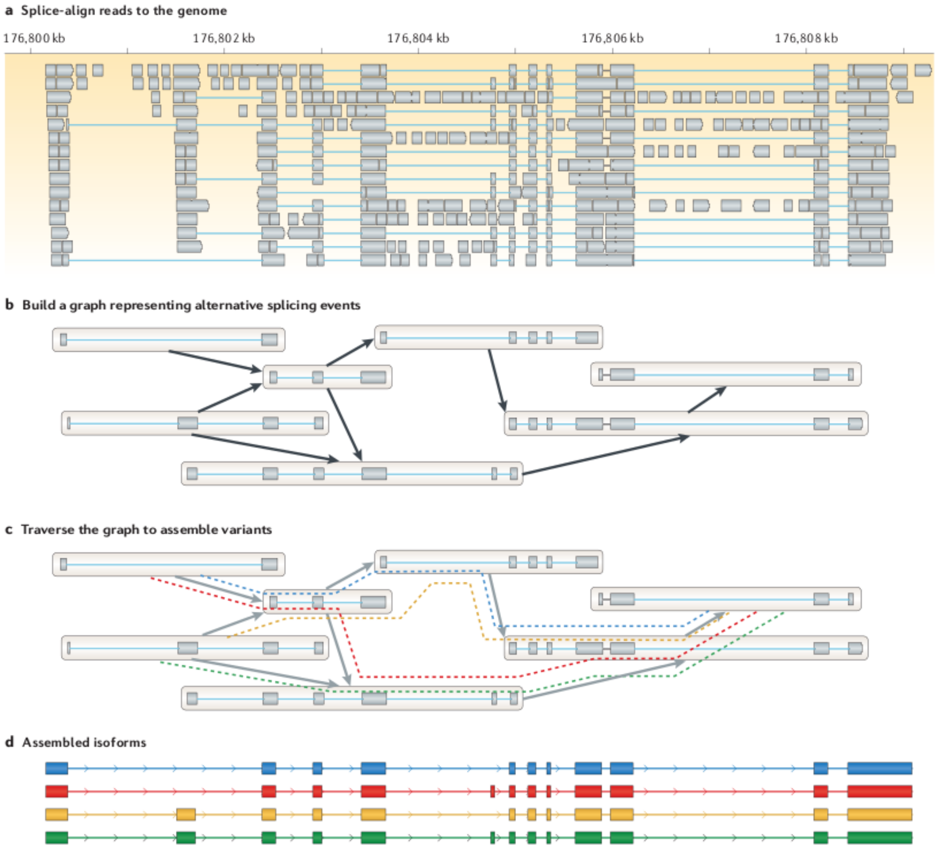

a Splice-align reads to the genome
176,800 kb
176,802 kb
176,804 kb
176,806 kb
176,808 kb
b Build a graph representing alternative splicing events
c Traverse the graph to assemble variants
d Assembled isoforms

Figure 3.1: Reference-based transcriptome assembly -

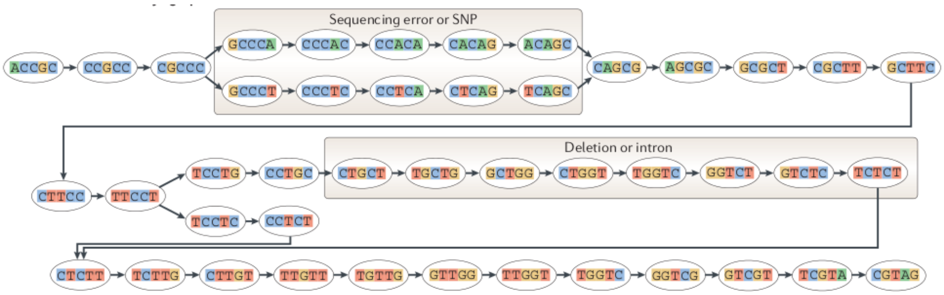


Figure 3.2: *De Bruijn* graph -

introduce the graph structure briefly. Then we compare the strategies employed by leading assemblers Trans-ABySS (106), Trinity (107), Oases (108), IDBA-Tran (103) and SOAPdenovo-Trans (104). Meanwhile, we analyse the advantages and disadvantages of them.

3.2.1 *De Bruijn* Graph

de Bruijn graph was originally invented by the Dutch mathematician Nicolass de Bruijn to solve the ”superstring problem” (109). It was firstly brought to bioinformatics in 1989 to assemble k-mers generated by sequencing by hybridization (110). Here a *k-mer* means a sequence of characters with a length *k*.

In bioinformatics, *de Bruijn* graphs are applied for assembly applications. A *de Bruijn* graph is a directed graph where an edge represents a k-mer and a node is assigned a $(k-1)$-mer. In the graph, a node is directly connected to another if there exists a k-mer whose prefix is the $(k-1)$-mer of the former node and whose suffix is the latter (Figure 3.3).

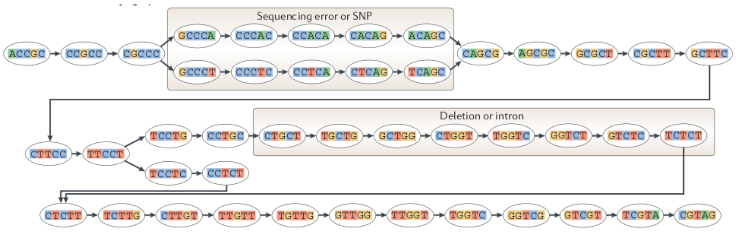


Figure 3.3: A sample *de Bruijn* graph - Right side is the *de Bruijn* graph. Here k value is 3. The sequences on the edges represent k-mers. Numbers on the edges indicates an Eulerian cycle , which produces a candidate circular genome ”ATGGCGTGCA”.

Every read is first broken into overlapping k-mers. For example, in Figure 3.3, the read ”CGTGCAA” is broken into k-mers ”CGT”, ”GTG”, ”TGC”, ”GCA” and ”CAA”. For all k-mers detected in the reads, nodes are created to model the connectivity. This process constructs a *de Bruijn* graph. Then the assembly problem is transformed to an equivalent problem of finding an Eulerian cycle (Eulerian path if the chromosome is linear). An Eulerian cycle is a path which visits all edges only once and ends at the node where it begins. Finding an Eulerian cycle allows one to reconstruct the genome by forming an alignment in which each successive k-mer is shifted by one position. It avoids computationally expensive tasks such as large volumn of pairwise alignments. An Eulerian cycle is a candidate of the original genome (102), as Figure 3.3 shows. For transcriptome assembly, an Eulerian path gives one candidate transcript.

Euler's theorem had proved that there must be an Eulerian cycle as long as we have located all k-mers present in the genome (102, 111).

In real assembly applications, *de Bruijn* graphs are customized to tackle potential problems. There are some hidden assumptions in a *de Bruijn* graph that are not held true for real datasets. For example, theoretically it is required that all k-mers present in the genome can be generated, all k-mers are error free and each k-mer appear at most once. However, sequencing error is very common for NGS projects. It introduces a large number of false nodes, resulting in a massive graph with millions of possible (mostly implausible) paths.

Modifications are applied to make *de Bruijn* graph applicable. A "read breaking" procedure helps to ensure that all k-mers appearing in the genome are detected (112). In 2001, an error correction strategy of the reads was applied before the real assembly process was started (100). The error correction is now commonly used. Later, an algorithm was proposed to remove short and noisy vertices from the *de Bruijn* graph efficiently. Meanwhile, to handle repeats, k-mer multiplicity, which indicates how many time a k-mer appears, was integrated into the graph.

Modern genome assemblers based on *de Bruijn* graph include EULER-SR (113), Velvet (89), ALLPATHS (88), ABySS (90) and SOAPdenovo (114). For comparison of these genome assemblers, please refer to (115, 116).

There are three major problems in the *de Bruijn* based approaches (117).

- Incorrect k-mers: sequencing errors will result in very complicated graph structure, from which the plausible paths are difficult to be determined.
- Gap problem: when k value is large or for those regions with low sequencing depths, some k-mers are missing.
- Branching problem: for repeat regions or highly similar transcripts, there maybe too many available branches. If k value is small, this problem becomes severe.

3.2.2 *De Novo* Transcriptome Assemblers

Table 3.1 is a list of state-of-art transcriptome assemblers. Within the eight *de novo* packages, "Multiple-k" is a general strategy which is employed by many assemblers. For example, Rnnotator (118) is a pipeline based on Velvet (89). Oases (108) is a transcriptome version of Velvet. It reports the transcripts that are merged from the resulting transcripts of multiple runs (with differnt k). Trans-ABySS runs ABySS multiple times for $26 \leq k \leq 50$ (106). IDBA-UD (119) focuses on a microbial environment for single cell sequencing. It also relies on the results of running Velvet with multiple k values. IDBA-Tran (103) starts from a lower k value (20) to build a noisy *de Bruijn* graph first, and then use the vertices as input to build a cleaner graph with a longer k. This strategy works to prune false connections in the graph.

Assembler	***De novo?***	**Parallelism**	**Support paired-end reads?**	**Support stranded reads?**	**Support multiple insert size?**	**output transcript counts**	**Reference**
G-Mo.R-Se	No	None	No	No	No	No	(120)
Cufflinks	No	MP	Yes	Yes	Yes	Yes	(9)
Scripture	No	None	Yes	Yes	Yes	Yes	(95)
ERANGE	No	None	Yes	Yes	Yes	Yes	(68)
IsoLasso	Yes	None	Yes	No	No	No	(121)
Multiple-k	Yes	None	Yes	Yes	Yes	No	(122)
Rnnotator	Yes	MPI	Yes	Yes	Yes	Yes	(118)
IDBA-UD	Yes	MP	Yes	Yes	Yes	Yes	(119)
IDBA-Tran	Yes	MP	Yes	Yes	Yes	Yes	(103)
Trans-ABySS	Yes	MPI	Yes	No	Yes	Yes	(106)
Trinity	Yes	MP	Yes	Yes	No	Yes	(107)
Oases	Yes	MP	Yes	Yes	Yes	no	(108)
SOAPdenovo-Trans	Yes	MP	Yes	Yes	Yes	no	(104)

Table 3.1: Comparison of current transcriptome assemblers. MP: multiple processor support; MPI: Message-passing interface support

Trans-ABySS (106), Trinity (107) and Oases (108) are all based on *de Bruijn* graph. Trans-ABySS and Oases derive from genome assemblers ABySS and Velvet respectively.

From our observations, roughly speaking, the assembly process can be divided into four major steps: error detection/correction, graph construction, and transcripts determination. In following paragraphs, we are going to compare the strategies employed by current assemblers (mainly ABySS, Trinity and Oases) for every step.

3.2.2.1 Error Detection/Correction

As we mentioned in the previous section, sequencing errors will result in complex graph structure. Since the first error correction algorithm was proposed in 2001 (100), error detection/correction is now a common step of assemblers. Preprocessing the raw reads reduces the variation of gene coverage while improving the computational performance of the assembly.

Almost all researchers chose to use k-mer frequency to filter out reads that contain sequencing errors (88, 91, 106, 107, 118). The rationale behind is that if a read contains some sequencing errors, its k-mers would appear in the RNA-seq library for much less times. First of all, all reads are broken into k-mers. Then the occurrences of these k-mers are counted and ordered. k-mers with lower multiplicity than some threshold will be marked as error. Rnnotator removes the duplicate reads at the same time. Trans-ABySS performs the error removal after the graph has been constructed. And IDBA-UD used multiple depth relative thresholds to remove erroneous k-mers in both low-depth and high-depth regions. Trinity removes those k-mers that are $<5\%$ abundant as compared with the most highly abundant k-mers of the group. It also identifies the seed k-mers with higher information content (Shannon's Entropy (123)).

The methods to deal with errors are relatively similar. The basic idea is to bias towards k-mers with higher frequency.

3.2.2.2 Graph Construction

Oases and Trans-ABySS don't construct the *de Bruijn* graph directly. They make use of the resulting contigs from Velvet and ABySS. Another common strategy is that they both run with multiple k values and merge the contigs to form a basis for further processing. Some internal algorithms are altered or abandoned because of the different characteristics of genome assembly and transcriptome assembly. For example, the Pebber and Rock Band algorithms from Velvet are not used in Oases since they assume uniform coverage across the genome.

IDBA-Trans goes further to implement a more sophisticated approach. They build and improve the *de Bruijn* graphs gradually with increasing k values. The contigs from previous iteration is applied as input to construct the graph in next iteration. IDBA-Trans is especially good at detecting and correcting erroneous branches.

After the set of contigs are obtained, Oases corrects the contigs with a set of dynamic filters (similar to *TourBus*) and static filters (remove contigs with low coverage). While Trans-ABySS merges the resulting contigs by utilizing the alignment tool BLAT (78).

Trinity uses a different approach. Its Inchworm module first assembles reads into unique sequences of transcripts using a greedy k-mer-based approach. A k-mer dictionary is created like other assemblers. Inchworm starts from the most frequent k-mers. Within those k-mers which share a $(k-1)$-mer with it, the one with largest frequency is chosen. This process iterates until it cannot be extended further. Inchworm reports a set of contigs which are unique and frequent.

Second utility of Trinity called Chrysalis then clusters Inchworm contigs into sets of connected components, and constructs complete *de Bruijn* graphs for each component. The concept of "component" is similar to "loci" of Oases. Ideally, all transcripts from one gene should be assembled into a connected component of contigs. But in real applications, due to sequencing errors, repeat patterns and common sequence patterns, a loci/component sometimes represents fragments of genes, or clusters of homologous sequences. Chrysalis groups the components by checking the overlapping between contigs and the reads spanning the junction across both contigs. Then it constructs *de Bruijn* graphs for each component with k value.

After the graph is built, these tools usually traverse the graph multiple times to simplify the structure. For example, Butterfly from Trinity merges consecutive nodes in linear paths to form a longer sequence, and it also removes edges that represent minor variants.

3.2.2.3 Transcripts Determination

The algorithms from Trinity and Oases to report transcrips are similar. The edges in the *de Bruijn* graph are weighted based on the k-mer frequency from the original set of reads. A dynamic programming algorithm is then applied to find those paths with higher scores. By the help of read pairs, they reduce the combinatorial paths to a smaller number.

Every plausible path is reported as a transcript by Trinity. But Oases goes further to merge transcripts by Oases-M, which runs Oases multiple times with a set of k values. The resulting transcripts are then combined to build another *de Bruijn* graph with another parameter k_{MERGE}. Then this *de Bruijn* graph is processed similarly to report the final transcripts. Graph merging itself is a complicated problem (103). The implementation of Oases is more difficult than others. IDBA-Tran contributes by introducing different pruning thresholds for the components. To distinguish the lowly expressed transcripts with the segments resulted from high sequencing errors, IDBA-Tran adopts a statistic module to determine a specific threshold δ for every component, which filters out the erroneous branches in the graph. Since the value δ is derived from the read coverage within the single component, it is more accurate.

We did not find any post-processing steps from Trans-ABySS.

To draw a conclusion, most of the modern *de novo* transcriptome assemblers are based on *de Bruijn* graph. The graph theory ensures the efficiency and correctness of the algorithms. However, they suffer from the potential problems in practical applications as we mentioned in Section 3.2.1. These tools suffer severely from sequencing errors, low-expressed genes and repeats. Although researchers tried to tackle these problems by various sophisticated algorithms, they result in complicated graph processing procedures and transcript differentiation mechanism.

Moreover, we observe that for paired-end RNA-seq libraries, these assemblers are not able to make use of the paired-end information until the Transcripts Determination stage because the reads are broken into k-mers in the very beginning. We can only find one study in the literature that used paired-end information when constructing the *de Bruijn* graph (124). But it was only a prototype and no experiments were conducted. EBARDenovo employs a similar strategy to PETA, i.e., use paired-end reads to help extend the low coverage regions. However, it does not build a graph structure to resolve the complexity of transcriptome, making it inadequate for a general transcriptome assembly application.

Some research groups had compared the existing tools based on some practical datasets (125, 126). But they lacked of standard evaluation metrics. In this study we use a set of evaluation metrics proposed by (6). These measures are expected to be better because they evaluate the performance from various perspective. We will give clear definitions of these metrics in Section 8.

In this study, we return to the traditional overlap-layout-consensus scheme. Our contributions are the full utilization of paired-end information during the assembly. With paired-end reads, we are able to construct a much cleaner graph from the very beginning. In next chapter, we demonstrates the strategies of our *de novo* transcriptome assembler PETA.

4

Problem Statement

4.1 *De Novo* Transcriptome Assembly

The transcriptome reflects the genes that are being actively expressed at the given time. Studying the transcriptome is necessary to understand the processes of cellular differentiation (127), molecular mechanisms controlling early embryonic development (128) and the underlying mechanism of diseases like cancer (129, 130). The transcriptome assembly problem is to identify the full set of transcripts, including large and small RNAs, novel transcripts from unannotated genes, splicing isoforms and gene-fusion transcripts (65, 131).

Clearly, exons from different variants of the same gene may contain identical exon fragments. In the *de novo* assembly, we are blind to the boundaries of exons and other regions. So we are more generally interested in common parts of the transcript variants rather than simply in common exons. Similar to (131), we define the concept of a *block*.

Definition 1. *A block is a maximal sequence of adjacent exons or exon fragments that always appear together in a set of transcript variants.*

According to this definition, a variant can be represented by a sequence of blocks. A block doesn't necessarily associate with exon structure. It either belongs to a variant completely, or is skipped. Figure 4.1 illustrates how a block is inferred from the variants.

Four blocks A, B, C and D are created from the figure. Between two consec-

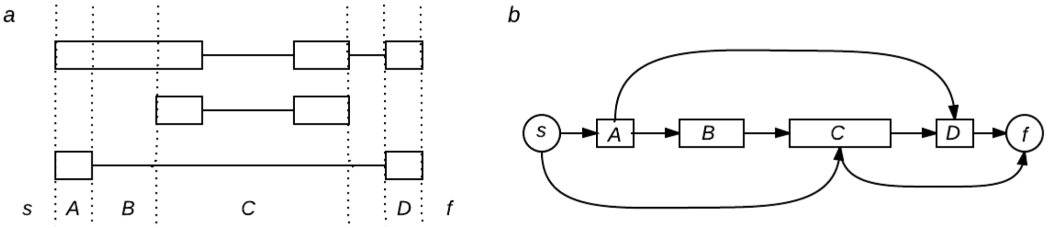


Figure 4.1: Block definition - The left part is a diagram of a gene with three variants. In our diagrams, rectangles represent exons or exon fragments, and horizontal solid lines are the intervening parts that are not contained in a particular variant (introns). Vertical dashed lines defines four blocks A, B, C, and D. The three variants can thus be described by the sequences $ABCD$, C, and AD. The corresponding splicing graph is drawn in the right part where vertices are blocks and edges are block junctions pointing in the direction of transcription, from 5' end to 3' end. The graph is completed by two additional vertices s (as the starting vertex) and f (as the final vertex). Vertex s is connected to all first blocks of the variants, and vertex f is connected to all last blocks.

utive blocks, a *block junction* represents the connection between them. Under this definition, a block can be an exon-intron-exon structure like C, or could be some portion of an exon like A. A block junction may across two exon segments that are far way, such as AD.

Given a set of variants S_0, a *splicing graph* is a directed acyclic graph whose vertices are blocks and edges are the block junctions. As long as there exists some block junction between two blocks, an edge is added to the splicing graph. Additionally, a starting vertex s and a final vertex f are added to the graph. Given two adjacent vertices u and v, an edge e connecting from u to v is represented as (u, v). The formal definition of a splicing graph is:

Definition 2. *A splicing graph G is a directed acyclic graph (V, E), where V represents the set of blocks and E the set of edges such that* $E \subseteq \{\{u, v\} : u, v \in V\}$

We can infer limited number of directed *paths* as we traverse the splicing graph. A *path* must starts from the vertex s and ends at the vertex f, and it consists of a subset of vertices and edges of from the splicing graph. A variant corresponds to a path or a continuous subsequence of a path. For example, the first transcript variant in Figure 4.1 is captured by a path *sABCDf*. However, a path from the vertex s to the vertex f doesn't necessarily correspond to any expressed variants.

For example, the annotated transcripts is $S_0 = ABCD, C, AD$, but the paths *sCDf* and ABC are also contained in the graph. Theoretically, if there are N blocks in the splicing graph, there are $2^N - 1$ possible paths.

We now formulate the problem of *de novo transcriptome assembly* as follows:

Problem Statement. *Given a set $S = \{x_1, ..., x_k\}$ of candidate variants which are paths from the splicing graph, and a set of constraints $\{C_1, ...C_m\}$, each indicating the total abundance of a subset of variants of S, report a subset of S such that all constraints are best satisfied.*

In our application, a constraint C_j reflects the number of sequence reads mapping to a particular block junction j, called the abundance of the block junction, and the corresponding subset of variants will be all variants of S that contain junction j. Figure 4.2 illustrates an example about how to define the constraints.

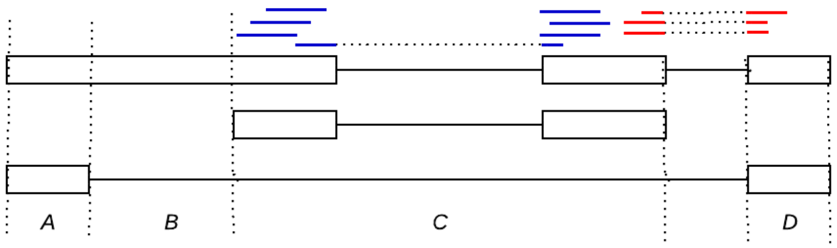


Figure 4.2: Constraints on the paths - The raw reads are mapped to the graph, including the block junctions. In the figure above, there are 8 blue reads assigned to the block *C*. And the 3 red reads are assigned to the block junction (*C, D*).

4.2 PETA: Paired-End Transcriptome Assembly

Started from next Chapter, we describe the details of our *de novo* transcriptome assembler called PETA (Paired End Transcriptome Assembler).

Due to sequencing errors and repeat patterns in the transcriptome, a *de Bruijn* graph contains a lot of false connections, resulting a very complicated graph structure (132). The success of a *de Bruijn* graph assembler relies on sophisticated error detection and correction afterwards. For example, Trinity abandons a connection if there are not enough k-mers to support it (5%). IDBA-Tran breaks the graph into components and then defines specific thresholds based on some statistical model. In addition, existing assemblers require a critical parameter k, which is the k-mer length for extension. A longer k ensures longer transcripts, while a shorter k gives higher sensitivity. To achieve a good trade-off between specificity and sensitivity, many assemblers, such as Oases and IDBA-Tran, accept multiple k values and merge the graphs or transcripts later. But it introduces more complexity for implementation.

We claim that full utilization of raw reads and paired-end information is able to construct a cleaner splicing graph and provide more accurate and reliable transcriptome. PETA follows the traditional overlap-layout-consensus scheme. It maintains a *pool* of reads to get next consensus base and extends the template base by base. It tackles the above issues by fully utilizing paired reads to help extend the templates, merge templates and validate graph paths.

In the following sections, we introduce preliminary observations on real RNA-seq libraries. First of all, we specify the definitions and notation to avoid ambiguity. Secondly, since two real RNA-seq datasets are used throughout the whole study, we first describe the *S.pombe* and Human RNA-seq libraries as well as the annotation transcripts used for the evaluations. From the real data, we investigate the usefulness of paired-end information in RNA-seq reads. In Section 4.6 we do some study to determine the key parameter L which is the minimal overlapping length between adjacent reads.

4.3 Definitions and Notation

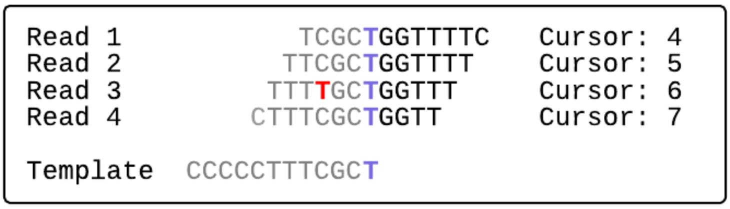
To avoid confusion, we describe the specific meaning of the terms/concepts we are using in this study. You can also turn to Section *Glossary* for a quick reference.

Four possible nucleotides are encoded as two binary digits as follows:

$$f(A) = 00_2,\ f(C) = 01_2, f(G) = 10_2, f(T) = 11_2 \tag{4.1}$$

We rely heavily on the paired-end reads. In a paired-end RNA-seq library, the two in-paired reads are called the *mate* (or *mate read*) of each other.

During assembly, a *template* is a nucleotide sequence being extended base by base from a *pool* of reads. As PETA proceeds, the template grows longer and longer. The pool is maintained on the fly, whose reads overlap at least with the template *tail* for L bases, which is a user defined parameter. For each read in the pool, a *cursor* value pointing to the next candidate nucleotide is maintained. An integer value *weight* is assigned to each read based on three features: overlapping length with the template, number of mismatches on the overlapping region and whether the mate of the read has been used by the template. After extending a template by one base, the cursor values on every read of the pool will be updated accordingly. An example is given as Figure 4.3.



```
Read 1           TCGCTGGTTTTC    Cursor: 4
Read 2          TTCGCTGGTTTT     Cursor: 5
Read 3         TTTTGCTGGTTT      Cursor: 6
Read 4        CTTTCGCTGGTT       Cursor: 7

Template  CCCCCTTTCGCT
```

Figure 4.3: Pool and cursor - Reads 1-4 are in the pool. Every read in the pool overlaps with the template for at least L bases. The reads are laid based on the cursor values. In this particular case, the next base should be *T*. If there are more than one possible base, we pick the one with heaviest weight value.

Since some transcripts share common segments, there are *connections* among the templates. A *connection* between two templates represents a probable block junction between two hidden blocks, which are portions of the two templates. A connection connects either end of the *branch template* to some locus (usually in the middle) on the *main template*. For instance, in Figure 4.4a, the template above is the branch template, and the one below is the main template. There are two connections between them.

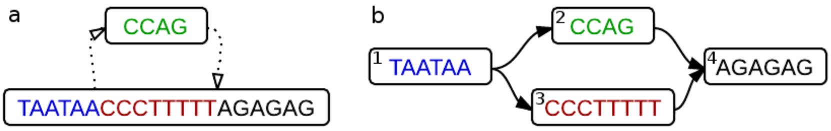


Figure 4.4: Connections between templates - In the left figure, there are two template connections. The sequences in the same color define four blocks. The left connection defines a block junction between the Block 1 and Block 2. And the right connection defines a block junction between Block 3 and Block 4. The corresponding splicing graph is shown in the right figure.

From the templates and all connections among them, the splicing graph can be constructed. We will describe the detailed implementation in the coming chapters.

4.4 Real Datasets

In order to deal with transcriptomes with different levels of complexity, we select two RNA-seq datasets that are well studied. The first one is a simpler transcriptome with moderate transcript variants. While the second one is complex human transcriptome. The annotated transcripts are used as reference. The aim of the assembly is to reconstruct as many full-length transcripts as possible. Meanwhile, longer transcripts are preferred.

The first dataset is from *Schizosaccharomyces pombe*, which is sequenced and prepared by (133). *S.pombe*, also called "fission yeast", is a species of yeast. It is used as a model organism in molecular and cell biology. It is a unicellular eukaryote, whose cells are rod-shaped. The transcriptome of this dataset is well annotated by Broad Institute, which is downloaded as the reference transcripts [1].

The second dataset SRX011545 is from human genome [2]. The dataset is used by Oases (108) as well. The annotated transcripts are downloaded from Ensembl database [3].

Statistics of the two dataset are listed in Table 4.1.

4.5 Useful Paired-end Information

Our assembler fully utilizes the paired-end information to get longer and more reliable transcripts. An important hypothesis is that the paired reads are of high quality: both reads of a pair actually origin from a unique transcript variant and the distance between them is within the correct range.

To validate this hypothesis, we align all RNA-seq reads on to the set of annotated transcripts using BWA (paired-end mode) (77). A pair is claimed to be a good one if:

- Both reads are aligned to the same transcript.
- The alignment directions of the two reads are the same.

[1] Broad Institute: http://www.broadinstitute.org

[2] http://www.ebi.ac.uk/ena/data/view/SRX011545

[3] http://asia.ensembl.org/index.html

	S.pombe	**Human**
Platform	Illumina	Illumina
Source	ENA	ENA
Study ID	SRX040570	SRX011545
Paired	Yes	Yes
Strand specific	Yes	No
Read length	68bp	45bp
# of reads	18,353,817 * 2	23,458,222 * 2
Mean insert size	326bp	200bp
Standard deviation of insert size	78bp	62bp
File size	2.7Gb	2.5Gb
Hashtabe size	3.5Gb	4.5Gb

Table 4.1: Dataset

- The distance between the two reads are within the correct range. The range is [*(insert size - 2.5 * standard deviation), (insert size + 2.5 * standard deviation)*].

Based on this role, we observe that the good read pairs for *S.pombe* and human datasets are 95% and 90% respectively. These numbers do not include the reads with two many mismatches. So the real good pairs are even more. The numbers indicate that, instead of being located at multiple positions for single reads, majority of the paired-end reads can be uniquely located to the transcript variants.

4.6 Determine the Overlapping Length

Since we are using classic approach overlap-layout-consensus to extend the templates, the first step is to get the reads that overlap with the template tail, whose length L is a key parameter specified by the users. It is similar to the length of k-mers for assemblers Trinity and IDBA-Tran. Larger value promises better specificity, but the sensitivity is scarified to some extend. While too short overlapping length will create too many connections among the templates, making a complex splicing graph.

To provide a feasible L value to adjust the tradeoff, we align the RNA-seq reads to the annotated transcript and check the least overlapping length between adjacent reads. Here we use Blat (78) to perform the single-end alignment. Because instead of reporting the best hit by BWA, Blat reports more alignments if a read is mapped to multiple locations. On every annotated transcript, we order the alignment hits by the mapping locations in a increasing order. Then the overlapping length between adjacent hits are counted. We finally collect the numbers of hits for every possible overlapping length. For example, for *S.pombe* dataset, there are 769 locations where the two adjacent reads overlap for 25bp.

The results show that, for *S.pombe* dataset, there are only accumulatively 24,348 reads (out of totally 34 million reads) that overlap with its adjacent hit for less than 26bp. The same number for human dataset is 693,544. We can conclude that for *S.pombe*, 25bp is a feasible number (<1%), which hopefully to assembly continuous transcripts. However for the human dataset, more than 15% of the reads overlap with adjacent reads for at most 25bp. Considering that the size of human transcriptome is much larger, and there are 254,555 reads show empty overlap with their adjacent reads, we conclude that the human dataset is more noisy.

We finally set 25bp as the default L value, because a lower value will largely increase the number of reads in the pool.

4.7 PETA

4.7.1 Implementations

PETA is hosted at http://caishaojiang.com/peta, where the source codes and user manual can be found. It is mainly implemented in programming languages C/C++. It can be run only on a 64bit Linux-like operating system, such as Ubuntu, CentOS and Fedora. Python scripts for data preparation and evaluation are also included in the package. Currently multi-threading feature is implemented only at the hashing step.

4.7.2 Workflow

Figure 4.5 is an overview of the PETA workflow. PETA consists of three major steps. First of all, a hash table is built from the raw RNA-seq reads. The hash table does not include the sequences of reads, but a list of k-mer occurrences on the reads. It is efficient to build a hash table. We spend 15 minutes and 23 minutes for the *S.pombe* and human datasets respectively. The details are explained in Chapter 5.

Then we start linear extension from high abundance reads. Paired-end information is utilized to ensure that we are assemblying longer and reliable templates. After all reads are consumed, we obtain a list of disconnected templates with reads assigned to them. Based on the overlap between templates and the paired reads spanning on the templates, we merge and connect certain templates to make them ready for constructing the splicing graph. These processes are described in Chapter 6.

Chapter 7 constructs the splicing graph, removes cycles in the graph and performs an expectation-maximization algorithm to report a final list of validated transcripts. In Chapter 8, we show the evaluation criteria, experiments design and the assembly results of different *de novo* assemblers. We also analyze the advantages and disadvantages of PETA. Some failed cases are analyzed in detailed.

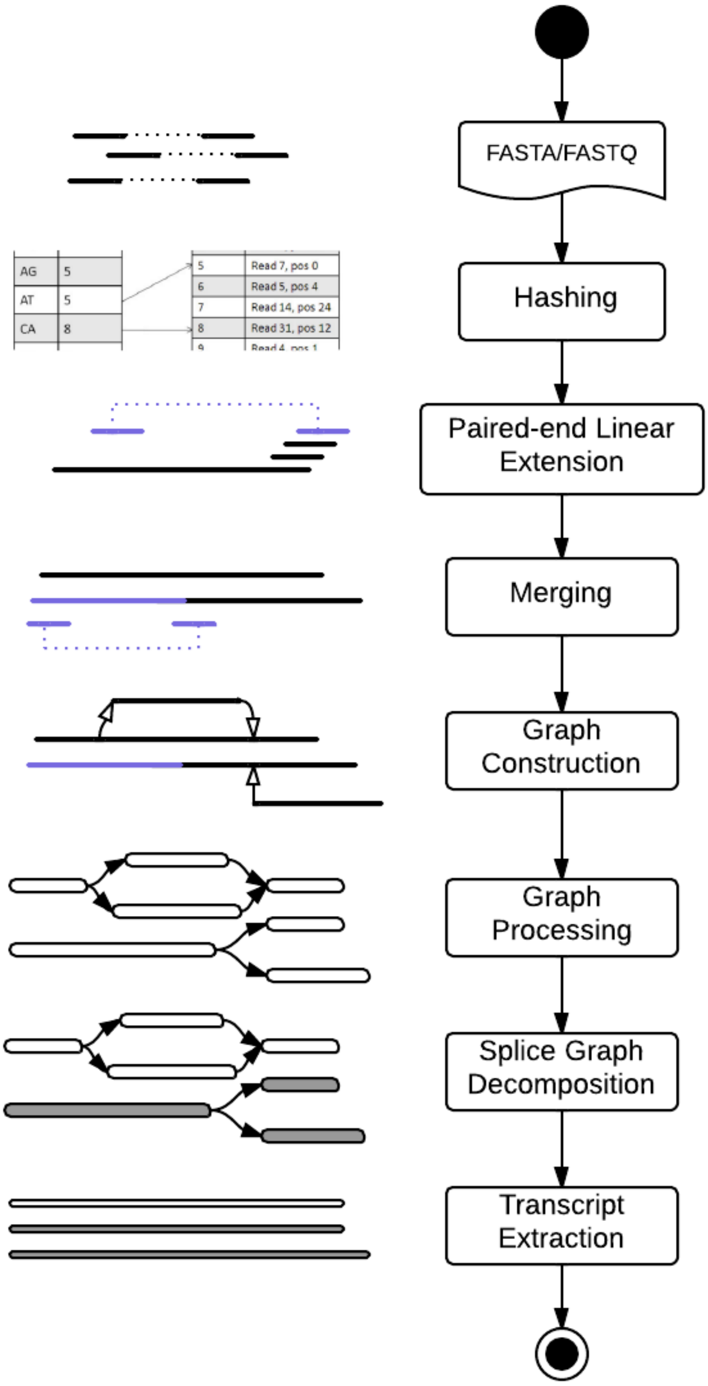


Figure 4.5: PETA workflow - The rectangles and longer lines in the figure represent templates, vertices and final transcripts. The short lines represent single-end or paired-end reads. The program is executed linearly, following the process from top to bottom

5

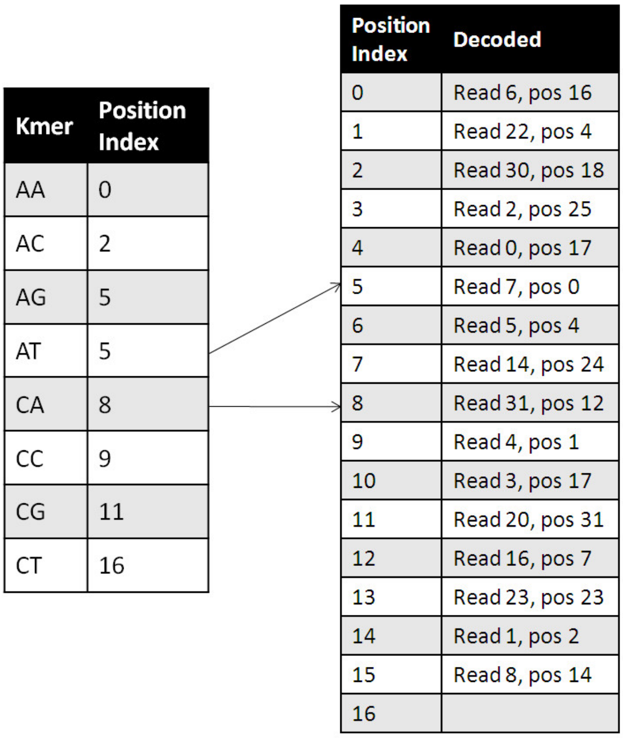
Hashing

In PETA, pairwise alignment is an essential component to align the template tail to the whole RNA-seq library to select those reads containing the tail. The alignment is implemented by a hash table approach. In this section, we introduce the process to build a hash table from the raw RNA-seq reads. We also explain every parameter to feed in to the hashing problem. Then we show how the pairwise alignment is performed. Finally, we also discuss the accuracy and limitation of our hashing strategy.

5.1 Build a Hash Table

PETA performs mapping by searching for k-mers occurrences in the reads first. A k-mer is some length-k substring from a RNA-seq read. It is used as a key in the hashtable to index the RNA-seq reads. A longer k-mer is more specific and is less likely to have collision in the hashtable, while it will occupy more space. A short k-mer is more sensitive. However, there will be more noise hits. With length k, there are 4^k possible combinations of k-mers. Currently we set the k-mer length to 11 by default.

Hashing in PETA is based on the approach introduced by SSAHA (134). The details are illustrated in Figure 5.1. The size of the array *k-mer* is 4^k. The size of the array *pos* depends on the number of k-mers to be hashed. PETA hashes a fixed number of k-mers for every read. It is determined by a set of user parameters as illustrated in Figure 5.2. PETA splits every read into *blocks*. For each block, PETA



Kmer	Position Index
AA	0
AC	2
AG	5
AT	5
CA	8
CC	9
CG	11
CT	16

Position Index	Decoded
0	Read 6, pos 16
1	Read 22, pos 4
2	Read 30, pos 18
3	Read 2, pos 25
4	Read 0, pos 17
5	Read 7, pos 0
6	Read 5, pos 4
7	Read 14, pos 24
8	Read 31, pos 12
9	Read 4, pos 1
10	Read 3, pos 17
11	Read 20, pos 31
12	Read 16, pos 7
13	Read 23, pos 23
14	Read 1, pos 2
15	Read 8, pos 14
16	

Figure 5.1: k-mer searching of SSAHA hashing strategy - Two one-dimension arrays *k-mer* and *pos* are created. Values of *k-mer* are indexes of *pos* array. Array *pos* maintains occurrences of k-mers in RNA-seq reads. In the example above, two arrows point out the occurrences of k-mer "AT". The hashtable has three *hits* for the k-mer, which appears in read 7, 5 and 14 at the positions of 0, 4 and 24 respectively.

always hashes 2 k-mers, which start with the first and second letters of the block. Another parameter is "interleaving size" i. It means that "for every i nucleotides, PETA will pick one to hash". For example, a read is "ACGTA", if i equals 2, the first, third and fifth nucleotides are picked to form a k-mer to hash, i.e., "AGA". It avoids too many hits for some highly repetitive patterns such as continuous A's.

Generally speaking, the number of k-mers from one read is calculated by:

$$Min((l - ki), 2b, (\frac{l - ki}{s} + 1) * 2) \tag{5.1}$$

Where l is the read length, k is the k-mer length, b is the number of blocks, s is the block size, and i is the interleaving size. So the storage cost for the hashtable is:

$$ht\_size = (4^k + n\_k - mer * N) * u \tag{5.2}$$

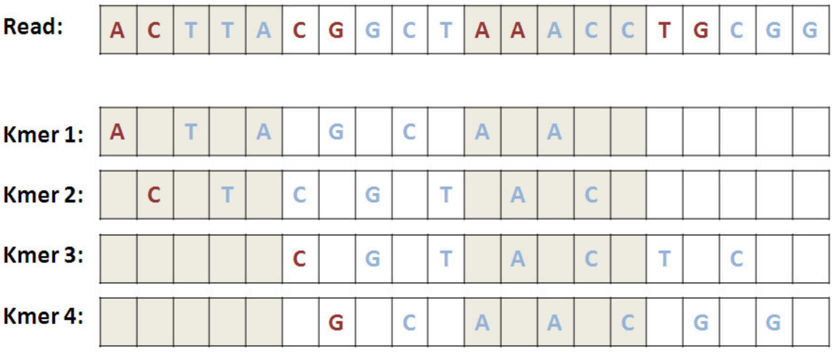


Figure 5.2: Determine k-mers to hash - A read is divided into 4 blocks (differentiated by background colors). Letters in read color are starting position of the k-mers. Interleaving size is set to 2. k-mer length is 7 for illustration. In conclusion, PETA will hash 4 k-mers from this read.

Where N is the total number of RNA-seq reads, and u represents the size of an integer. PETA supports 64-bit machines only, so u is 8 bytes by default. It is recommended to set the parameters such that $b * s$ roughly equals the read length l, such that the whole read is covered. Meanwhile, in order not to make the hashtable too large, 8 to 10 k-mers are recommended to hash for each read. In our experiments, the size of hashtable is 1.1 to 1.5 times of the RNA-seq library size.

The hashing step reads in and processes the sequences chunk by chunk. So the memory usage is relatively consistent regardless how large the library is. Of course, large libraries take longer time. When performing alignment, the original reads are required to be loaded as well.

5.2 Pairwise Alignment

With the hashtable, we are able to find the reads which contain a particular k-mer. Next we are going to discuss how to perform pairwise alignment for a tail. Mismatches are allowed during this process.

The first step is to identify all reads that align to the tail of the template allowing at most δ mismatches (by default δ is 2). To speed up the process, we use hashing as follows. First, all possible k-mers in the tail are obtained using the same parameters "k-mer length k" and "interleaving size i". For example, assume k is 4 and i is 2. Then a tail "ACGTACGT" has k-mers "AGAG" and "CTCT". The length of a tail is specified by the users. Please note that this value should not be smaller than $(k * i - 1)$. Otherwise, PETA could not obtain any k-mer from the read. For a tail with length t, there are $(t - k * i + 1)$ k-mers.

PETA then searches for these k-mers in the hashtable as shown in Figure 5.1. A hit corresponds to a read which contains the particular k-mer, but it doesn't mean that the tail is a substring of this read. Hits of all k-mers are combined together. PETA iterates through all reads to conduct base-by-base matching. During the matching process, the number of mismatches are counted on the fly. All reads with more than δ mismatches are thrown away. Remaining reads are reported as the result the pairwise alignment. Introducing indels would make PETA spend much more time, here we do not support gap alignment.

Finally, the time complexity of pairwise alignment is analysed as followed. To align a tail onto the RNA-seq library, the time efficiency is linear to the number of hits. Searching in the hashtable takes expected constant time. To allow mismatches, we go through every hit base by base. Suppose there are h hits and the read length is l, the time efficiency to get qualified reads is $O(h * l + C)$, where C is some constant. For the tails which have frequent k-mers in the library, the number of hits would be large. Thus PETA runs slower when extending the high coverage regions.

Both strand specific and non-strand specific libraries can be processed by PETA. If the query is from a non-strand specific library, we first align the query, and then align the reverse complement of it. After the position value are modified accordingly, we combine the resulting hits together. So running PETA on non strand specific libraries doubles the time.

5.3 Accuracy and Limitations

Results of PETA's pairwise alignment are suboptimal. In this section, we discuss about the performance of our pairwise alignment algorithm. Meanwhile, we point out some limitations of it. It this section, an "incorrect" k-mer is some k-mer with mutated nucleotides.

Let's have a look at the case of no mismatches first. To obtain all qualified reads, we need to make sure that all qualified reads hash at least one k-mer that can be inferred from the tail. For a RNA-seq read, we hash two k-mers for every block (Refer to Figure 5.2. Block size is s), the longest substring without any hashed k-mer has length:

$$min\_t = (s - 2) + (k * i - 2) \tag{5.3}$$

Our cleaning algorithm is similar to the one in PE-Assembler (91). First, we calculate the k-mer frequency for each read. For instance, in Figure 5.2, the length of the longest substring without a hashed k-mer is 15. That is, the substring starting from position 2 (with length 15) has no k-mer hashed. Any substring with length longer than 15 has at least one k-mer hashed. So as long as the length of tail is longer than 15, all reads having this tail are guaranteed to be found.

This hashing-based pairwise alignment suffers from mismatches because every mismatch in the read will result in multiple wrong k-mers. PETA guarantees to find all reads allowing one mismatch under some conditions. Basically, we need to specify a longer tail.

$$min\_t = s + (k * i - 2) \tag{5.4}$$

For the example, in Figure 5.2, the value of min_t is 17. Imagine that the sixth nucleotide 'C' is mutated, then the k-mer 2 and k-mer 3 are contaminated. PETA has to rely on k-mer 1 or 4 to retrieve this read. So the longest substring without a correct hashed k-mer starts at position 1.

Our approache has some limitations. Since hashing strategy is suboptimal, not all positive hits are guaranteed. For example, if the read length is shorter than *2ki* and the interleaving size is 2, a continuous 2-base error in the middle of read will make this read not obtainable by PETA. In Figure 5.2, if 10th and 11th letters 'TA' are both mutated, all 4 k-mers would be polluted, because every k-mer in the read has one base error. If the error occurs in either end of the read, we would expect that at least some k-mers in the read are not affected. However, it does not affect the overall performance. Because when the read length is longer than $2ki$, the problem is gone. Even if the read length is as short as 35bp, we can disable the interleaving (set size to 1) and use continuous k-mers instead. Of course, in this case users need to consider the tradeoff on the sensitivity gain by spaced seeds and continuous seeds.

To draw a conclusion, our alignment module is able to get all reads with no more than one mismatches. And most of reads with two mismatches can be obtained. The time efficiency is $O(h * l + C)$.

6

Extension and Connection

In this Chapter, we introduce the procedure to assemble the raw reads into a list of disconnected templates. We claim that our templates are accurate and more likely to from continuous transcripts. Meanwhile, the connections are reliable and they capture the real splicing events. The underlying rational of our advantages are:

- The reads with longer overlap with the template are more reliable.
- The reads with paired support are more reliable.

Different from k-mer based assemblers such as Oases and Trinity, we use the raw reads to assemble contigs directly. We believe that the raw reads, which are usually longer than the k-mers, are more reliable. Although PETA requires a similar parameter k, this value is not so sensitive comparing with other applications. The main reason is that we are able to go through lowly expressed regions by utilizing the paired-end information. This is a significant advantage of PETA.

We first illustrate how to determine the starting reads. Then we describe the strategy to maintain the pool to extend a template. Finally we show the usage of paired reads to connect separated templates.

6.1 Starting Reads

Linear Extension extends a template from some starting read. But some reads are with low quality if:

- The read has some sequencing error.
- The read does not overlap with others.

To ensure that we start from some high-quality reads, we group the reads by mapping each other (allowing two mismatches). The mapping is performed by a multi-thread pairwise alignment process on the hashtable, so it is very efficient. For a RNA-seq library of 7.5 million reads, PETA takes around 3 minutes to finish the grouping (4 threads). A read with more similar reads is supposed to be from some highly expressed transcript, which is believed to be more reliable. Meanwhile, since we group by reads, which are usually much longer than L, it shows less bias towards short repeat patterns.

The raw reads are sorted decreasingly by the number of similar reads. PETA picks the most frequent reads to extend until all reads are used. This strategy guarantees to reconstruct those highly expressed transcripts. A read will be marked as USED if some template uses all bases from it. A USED read would not be considered for extension any more.

There are some reads remaining after the starting reads are consumed. These reads may be from lowly expressed genes. We then pick those reads with higher k-mer frequency as starting reads and repeat the iterations.

6.2 Linear Extension

The basic idea to extend a template is as illustrated in Figure 4.3. A template is extended base by base. During extension, a pool of reads are maintained on the fly. Each read in the pool maintains a value called *cursor*, which points to the next nucleotide contributed by the read. All nucleotides at the cursor determine a consensus character to append to the template in every iteration. A read is added to the pool if it overlaps with the template tail. It will stay in the pool until its cursor moves to the end, and then this read will be marked as USED.

To determine the next nucleotide, we assign different weights on the pool reads based on its overlapping length with the template, number of mismatches on the overlapped subsequence, and the paired-end support. For instance, in Figure 4.3, Read 4 is assigned a heavier weight comparing with Read 1, because the overlapping length between Read 4 and the template is longer. In addition, Read 3 has one mismatch with the template. Its weight is decreased by 1. In the pool, a read is allowed to have at most two mismatches. Otherwise, it is likely to origin from some junction, or it is because of sequencing errors. Such a read is removed from the pool and reset to be FRESH again. This strategy will 'squeeze' out the reads with erroneous bases and improve the accuracy.

De novo assemblers usually suffer from low coverage regions and sequencing errors. Low overage results in fragmented contigs, and sequencing errors introduce many false branches in the graph. Some assemblers propose to use a shorter k-mer. However, it introduces false connections between transcripts from different genes. Even worse, shorter k-mer results in a more complicated graph. We observe that a read is more reliable if its mate is already used by the same template before. So we utilize the paired reads to tackle this problem in two novel ways.

First of all, to determine the next nucleotide of the template, we assign a much heavier weight to the reads whose mates are already used by the same template before. This heuristic is clearly illustrated in Figure 6.1. In conclusion, the weight of a read in the pool is calculated by:

$$weight = (overlapping\_length - mismatches) * (paired\_weight) \tag{6.1}$$

Where *paired_weight* is by default 1000 if its mate is used by the same template. Otherwise, the value is 1. The heavy weight ensures that paired reads are always maintained on the same template. Such a template is more likely to be a continuous region from some transcript. The weights are given in Table 6.1.

Feature	**Score**
Overlapping	1 * (length of overlapped segments between the read and template)
Mismatches	-1 * (number of mismatches on the overlapped segments)
Mate support	1000

Table 6.1: Weights for Read Features

We also utilize the paired-end reads when the pool is empty. It helps to extend the regions with low coverage. In this case, we add a read to the pool if it satisfies three conditions: 1. Its mate is used by the template before; 2. If the read is added,

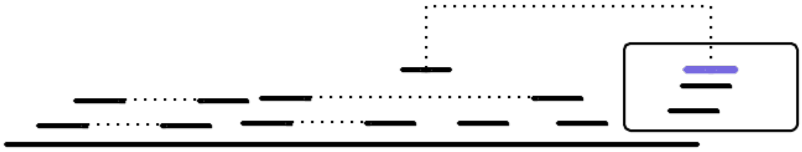


Figure 6.1: Weights in the pool - The short bars represent RNA-seq reads. Paired reads are connected by a dashed line. The longer bar at the bottom represents the template. The rectangle at the top-right corner is the current pool. There are three reads in the pool. The read in dark blue color is assigned a much heavier weight because its mate is used on the same template.

the distance between the read and its mate is within the correct range; 3. It overlaps with the template tail for at least 11bp. The *correct range* is defined as [*(insert size - 2.5 * standard deviation of insert size), (insert size + 2.5 * standard deviation of insert size)*] Although the overlapping length is as short as 11bp, we can use them for extension with high confidence. This strategy works well for transcripts longer than the insert size.

The reads on a template are marked as USED and would not be used for Read Extension any more. In the end, the templates do not share any segments longer than the tail length. Meanwhile, the template identity and the locus are stored as attributes of the USED reads.

The pesudocode is showed in Algorithm 1.

6.3 Template Merging

However, the complexity of transcriptome lies in various splicing patterns of the genes. It is very common that variances of a gene share long exonic segments.

In previous section, we obtain a list of disconnected templates, which are not supposed to share any segments longer than the read length. From our observations, although some transcripts indeed exist, they are fragmented into smaller portions whose overlapping length is short. To deal with these simple cases, we perform merging under some restrict thresholds. Figure 6.2 shows two templates that are supposed to be merged to form a longer template.

We categorize the reads on a template into three types. The first type is called

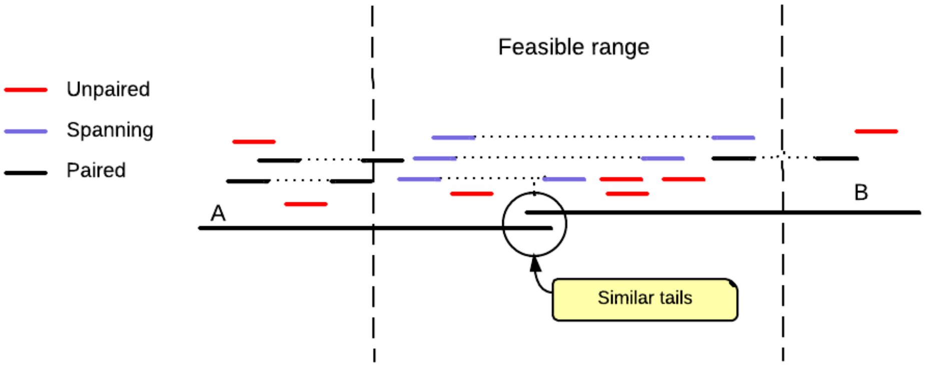


Figure 6.2: Merging templates - Two templates A and B are merged. The short lines represent the RNA-seq reads. There are three paired-end reads spanning A and B. The distances of the three pairs are within the feasible range. And the right end of template A and the left end of template B overlaps.

Paired, which means that the two reads of a pair are both mapped to the template, i.e., the black color reads in Figure 6.2. There also exist *Unpaired* reads (in red color and blue color) whose mates are not mapped to the template. It can be caused by high sequencing error on the mates or alternative splicing. The third type, a *Spanning* pair (in blue color), means that each read of a pair locates on a different template. And the distance between the spanning pairs should be within the feasible range.

When we attempt to merge two templates, first of all we define the *feasible range*, which is [*(insert size - 2.5 * standard deviation), (insert size + 2.5 * standard deviation)*] We would merge two templates if all of the following conditions are satisfied:

- The similarity score of the right end of template A and the left end of template B is at least 4.
- There are at least two spanning pairs.
- For all *Unpaired* reads within the feasible range, at least 50% of them are spanning pairs.

The merging strategy is conservative because the similarity score between two templates is allowed to be as low as 4. It means that 4bp overlapping is already acceptable. The similarity is obtained by a customized Smith-Waterman algorithm

(Algorithm 2). We believe that enough spanning reads indicate that the two templates are from the same transcript and should be merged. Our experiments show that this strategy adds around 350 full-length transcripts for *S.pombe*.

Since a transcript may be fragmented into more that two segments, we perform the merging iteratively until none of the templates can be merged.

To adapt to insertions and deletions, we implement a customized Smith-Waterman algorithm to compare the sequences. Since we are only interested in moderately similar cases, we allow only two indels and at most two mismatches. Generally, the implementation occupies $O(N)$ memory space. If the two sequences are not similar, it will stop after a few iterations. The running time for the worst cases are $O(N^2)$. Here N is the smaller length of the two sequences. The pesudocode is shown in Algorithm 2.

6.4 Template Connection

After merging, the templates become longer but stay unconnected. As we mention previously, alternative splicing makes the transcripts from the same gene share exonic segments. During Template Connection, we connect the templates by introducing block junctions between them. In this way, the blocks and block junctions are both defined. Two types of connection are performed.

Ideally, the existing templates should not share any regions longer than the read length. Because as long as a read is used by some template, it is "frozen" and would not be use by another template any more. In this case, if two transcript variants ABC and AC coexist, we would likely obtain two templates AC and dBe, where d and e are short segments from the junction reads, so d and e should be subsequence of AC.

Before connection, a small hash table is built for all 11-mers on all templates. The purpose of the template hash table is to efficiently identify those templates that share a subsequence with each other. The hashing strategy is exactly the same as the hash table for reads. The only difference is that the size of reads is much larger. The interleaving (Chapter 5) value for template hash table is 1.

The template hash table enables us to select those templates overlap with each other. If there are two segments shared by the two templates, we try to connect both ends of the branch templates. The strategy is illustrated in Figure 6.3.

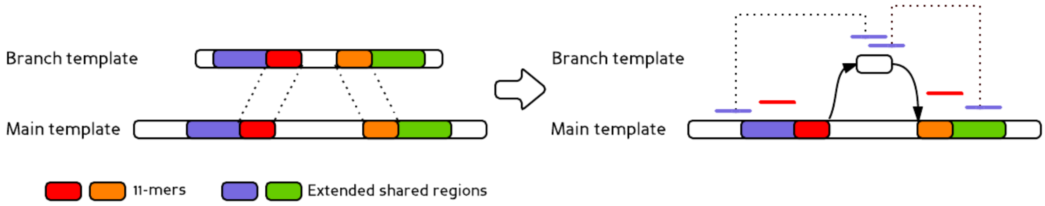


Figure 6.3: Both end connection - Connect both ends of the branch template to the main template. Similar to merging, the paired-end reads are used for conservative validation

From common 11-mers found on the two templates, we can easily discover the maximum segments shared by them (the blue+red and orange+blue regions in left part of Figure 6.3).

Then criteria to validate the connection are similar to merging. The difference is that we require there are spanning paired-end reads at both portion of the main template. In Figure 6.3, the branch template may be an exon that is contained in another transcript variant.

Sometimes only one end of a branch template can be merged. For example, two variants *ABC* and *ABD*. We likely to get two templates with sequences *ABC* and *D*. Then we need to connect left end of the later template to the former template. The criteria for one end connection is also the same as the both end connectioin.

The conservative manner creates the connections cautiously. That ensures all our edges in the splicing graph are supported by paired-end reads. From the *S.pombe* experiments, we can conclude that the connection strategy adds around 50 full-length transcripts in the results.

```
Input  : $Q$: a starting read
         $Hash$: hashtable of RNA-seq reads
Output: $T$: a template
TemplateExtension {
T = Q;  // Initialize the template;
$tail$ = $k$-length subsequence at the end of Q;
$pool$ = Empty;
while True do
    // Align the tail and add overlapped reads to the pool;
    $AlignTail(Hash, pool, tail)$;
    // The pool is empty, add overlapped reads with
    paired-end support;
    if $IsEmpty(pool)$ then $AddOverlapMates(pool)$;
    if $IsEmpty(pool)$ then break;
    // Assign weights to reads and get consensus base;
    $next\_base = DetermineNextBase(pool)$;
    $ExtendTail(tail, next\_base)$;
    $ExtendTemplate(T, next\_base)$;
    // Update the cursor value; remove reads from pool;
    $Forward(pool)$;
    // Remove reads with more than 2 mismatches from the
    pool;
    $RmHalfClipReads(pool)$;
end
}
```

Algorithm 1: Template Extension from a starting read Q

```
Input  : $A$: sequence A; $B$: sequence B
         $M$: minimum score
Output: $True$ if $A$ and $B$ are similar; $False$ otherwise
SmithWaterman {
$rows = Length(A)$;
$columns = Length(B)$;
$previous\_row = Integer[columns + 1]$;
$current\_row = Integer[columns + 1]$;
for $i = 1$ to $rows$ do
    for $j = 1$ to $columns$ do
        $up = previous\_row[j] + SCORE\_GAP$;
        $left = previous\_row[j - 1] + SCORE\_GAP$;
        if $A[j - 1] == B[i - 1]$ then
            $up\_left = previous\_row[j - 1] + SCORE\_MATCH$;
        end
        else
            $up\_left = previous\_row[j - 1] + SCORE\_MISMATCH$;
        end
        $current\_row[j] = Max(up, left, up\_left)$;
    end
    for $j = 1$ to $columns$ do
        $previous\_row[j] = current\_row[j]$;
        $max\_score =$
        $current\_row[j] > max\_score?current\_row[j] : max\_score$;
    end
    if (max_score + (rows - i)) * $SCORE\_MATCH < M$ then
    Return $False$;
end
Return $True$;
}
```

Algorithm 2: Customized Smith-Waterman algorithm for global alignment

7

Graph Processing

Until now, we have assembled a list of connected templates, where the connections are highly reliable because we use paired-end information for validation. However, from the Problem Statement section 4, we know that even if the block junctions are true, there could be still some paths that are not real transcript variants. In this chapter, we discuss how we construct and process the splicing graph. And finally we explain how the EM algorithm is applied to determine the valid set of transcripts.

7.1 Graph Construction

Figure 7.1 illustrates the workflow to break templates and add edges between the vertices. The templates with length 0bp (Template B) will be erased. And the junctions with same locus will be merged into one breaking point (Locus 917). In this example, template A consists of four blocks.

If we build the splicing graph from k-mers without validation, the graph structure would be more complicated due to sequencing errors and global common patterns. It is difficult to remove false connections given large amount of short vertices. Our splicing graph eliminates a lot of false connections from the very beginning.

The graph construction is intuitive. First of all, we identify and order the connection locus on a template. Every locus is a breaking point to break the template. For example, in Figure 7.1, Template A has three connection locus (locus 917 is duplicated). Each template is broken into unique continuous blocks, i.e., vertices

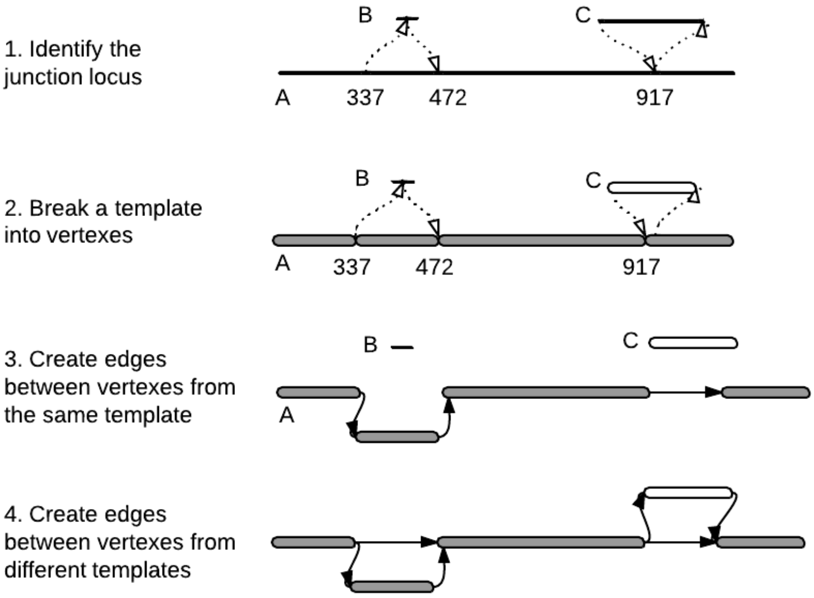

1. Identify the junction locus
B
C
A
337
472
917
2. Break a template into vertexes
B
C
A
337
472
917
3. Create edges between vertexes from the same template
B
C
A
4. Create edges between vertexes from different templates

Figure 7.1: Graph construction example - Template *B* is with length 0bp. It simply connects two portions of Template *A* tegother. For clear illustration, we draw a short bar as Template *B*. It is likely that Template *A* is constructed first. Later Template *B* and *C* are connected to Template *A*.

in the splicing graph. Nearby vertices from the same template have connections between them naturally. Lastly we create edges between vertices from different templates and form the splicing graph.

Let the number of vertices be V and the number of edges be E, the complexity to construct the graph is *O(V + E)*, since we visit every vertex and edge in linear time.

After construction, we apply some heuristics to simplify the splicing graph. Although the splicing graph is clean, there are probably short vertices that represent minor deviation (supported by comparatively few reads). For example, if a template connects to the end of another template, the last vertex from this template is short and should be removed.

More importantly, there may be cycles in the splicing graph. These cycles are caused by either repeat patterns or some sequencing errors. Since we extract some paths as transcripts, these cycles should be broken before determining the combinatorial paths. In graph theory, it is a classic problem called 'Detection of Strongly Connected Components (SCC)'. An SCC is a subgraph where every vertex is reachable from any of other vertices. There are many sophisticated solutions to tackle this problem. Tarjan's algorithm is a famous one (135). Its complexity is *O(V + E).* We implement Tarjan's algorithm and detect the SCCs in the splicing graph. For these SCCs, we simply remove some edges from it to break the cycles. This process is repeated iteratively until there is no cycle in the graph.

Until this step, we have constructed a clean and reliable splicing graph. Ideally, the transcripts from the same genes are closely located together in the graph. Look at the graph globally, we can find 'clusters' of vertices and edges. They have few connections between each other. These 'clusters' are named *components* by Oases, Trinity and IDBA-Tran. Instead of dealing with the whole graph, a better and common solution is to break the graph into smaller components and deal with the components more efficiently.

PETA uses the same strategy to process the splicing graph. Since PETA achieves a clean graph, the decomposition is not complicated. If it detects that two clusters are connected by very few edges, it removes the edges if there are not enough paired-end reads spanning these clusters. Disconnected subgraphs are then stored as components. From our experiments, the most complicated components contain around 200 vertices, which can be processed with moderate memory and running time.

7.2 EM Algorithm: Transcripts Extraction

With RNA-seq reads, the prediction of exons and splicing events are now resolvable. However, the reconstruction of full-length mRNA transcripts remains challenging, especially for genes with highly complex alternative splicing patterns.

This study does not include the quantification analysis. Our aim is to report all transcripts that are expressed. We apply a state-of-art statistical test to determine an optimal set of paths that explain the reads best. This approach is proposed by DiffSplice (136), a reference-based software to optimize differential transcription problem in the splicing graph. We construct the splicing graph in a *de novo* manner (previous sections), and then apply the DiffSplice Expectation-Maximization (EM) algorithm to determine the probability of component paths given the read coverage on them. Our contribution lies in applying the sophisticated statistical model to *de novo* assembly application successfully.

In this chapter, we first give an overview about existing approaches to tackle the problem. Then in section 7.2.2, we describe our implementation to adapt DiffSplice algorithm to our Transcript Extraction module.

7.2.1 Overview

To give an impression about the complexity, Figure 7.2 is the splicing graph of 7 human transcripts from the same gene ENSG00000174564. In the graph all splicing events are real. There are 35 combinatorial paths, within which only 7 of them cover the correct transcripts.

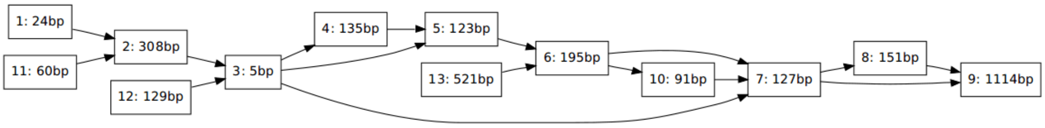


Figure 7.2: 7 transcripts from gene ENSG00000174564 - The graph is constructed from a simulated dataset. The labels in vertices are in the format of [vertex id: vertex length].

Existing approaches to extract full-length transcripts are categorized into three categories. The first category, like Cufflinks, performs transcripts inference and abundance estimation followed by differential test of relative abundance. This method is ideal, but it relies on accurate transcript quantification, which itself is a challenging problem. Although sophisticated techniques and/or statistical models are applied, the quantification of transcripts are 'unidentifiable' in some cases (136, 137).

The second strategy indirectly detects differential transcription by aggregating changes of multiple features (138, 139). For example, a statistical test called Maximum Mean Discrepancy is used to compare read coverage on all exons (138). Trinity develops a dynamic programming algorithm to append plausible edges to existing paths, with the help of read support and paired-end reads. But they are not able to optimize the global features. For example, Trinity only captures one full-length transcript from the example in Figure 7.2.

Approaches in last category examines the transcripts on annotated alternative splicing events in existing databases. They are proved to be reliable. But the disadvantage is that the performance replies on the annotation quality. We are working on *do novo* assembly of transcriptome, whose corresponding genome is usually not ready. So this approach is not suitable for this application.

DiffSplice proposed a state-of-art algorithm for transcript differentiation. It first maps the reads on to the reference genome to build a splicing graph. The splice graph is further decomposed to smaller units called *Alternative Splicing Modules* (ASMs). Then it applied the statistical test for every ASM to determine the expression levels of all paths.

7.2.2 Implementations

In statistics, an expectation-maximization (EM) algorithm is an iterative method for finding maximum likelihood of parameters in statistical models, where the model depends on unobserved latent variables (140).

DiffSplice is designed for reference-based applications, while we reimplement its EM algorithm in a *de novo* manner to assign expression abundance to candidate paths. Instead of mapping reads to the reference genome, we map the reads to our splicing graph (including vertices and edges). Our implementation is slightly different from the DiffSplice package, but the performance is promising based on our experiment results. For the example in Figure 7.2, our program is able to extract all 7 expressed transcripts exactly. Our experimental results also prove its capability for transcription determination.

We describe the statistical model of DiffSplice briefly (136).

Read coverage is the only feature used by the Expectation-Maximization (EM) algorithm. The aim is to get a subset of paths from all combinatorial paths, such that these paths will 'explain' the read coverage to fit in the ideal distribution optimally. On every vertex and edge, there are reads assigned to them. But RNA-seq reads are usually shorter than 100bp, we cannot distinguish where a read is really from. For instance, two transcripts share a long vertex. We don't know from which transcript the read origins. Every combinatorial path is assigned a probability, indicating how large the chance that this path is expressed.

Assume the sequencing procedure as a random sampling process, in which all reads are sampled independently and uniformly (141). When N_t, the number of reads from a path t, is large enough, we have:

$$C_{e|t} \sim N(C_t, \frac{r(l_t - l_e)C_t}{l_t l_e}) \tag{7.1}$$

Where $C_{e|t}$ represents the read coverage on exonic segment e from path t. C_t is read coverage on path t. And r denotes the length of a read. Both vertices and edges are treated as *segments* here, denoted as e. They are both continuous segments on paths. The length of edges l_e are decided by moving some bases from probable left and right vertices. From the equation, the $C_{e|t}$ values for different elements are unbiased for C_t. The variance of C_t varies according to the coverage C_t and the segment length l_e.

The length of an edge is not longer than $2 * (r - 1)$. This is slightly different from DiffSplice, whose edge length is the read length r. The reads at the junction areas are resided on the edges.

We get all combinatorial paths from a component. The number of paths may be large. So we validate and remove some paths from consideration. The rationale is that there must be some reads covering vertices that are shorter than read length. By 'covering', we mean that a read spans on the short vertex itself, as well as its previous and next vertex. This strategy will remove those false paths resulted from short common patterns.

From the distribution function, we can derive the likelihood function, which is maximized by EM algorithm. The likely function and the maximization process is the same as DiffSplice (136), so we do not include them in this paper.

The EM algorithm converges after some iterations. We stop the iterations when the probability change in two consecutive iterations is smaller than 0.000001 or it reaches the maximum iteration number 200000. The paths with a probability values larger than 2% are reported as the transcripts.

8

Experiments and Discussions

In the experiments, we compared the performance of PETA with state-of-art *de novo* transcriptome assemblers Oases, Trinity, Cufflinks and IDBA-Tran.

We use the evaluation metrics proposed by (6), which would be described clearly in Section 8.1. Transcripts reported by the assemblers are aligned to the annotated transcripts by Blat (78), which provides a good trade-off between accuracy and efficiency. The results are derived from the alignment hits.

The datasets we use for evaluation are described in Section 4.4. They are RNA-seq libraries from *S.pombe* and human with around 40 million reads.

The performance of PETA is comparable with other *de novo* assemblers. The content is organized as followed. We first give clear definitions and implications of the evaluation metrics. Then we show the evaluation results on the two real datasets. We also run the assemblers on a subset of the *S.pombe* dataset. The performance comparison shows that PETA is able to deal with RNA-seq with low abundance. Finally we show the experimental results and analyze the advantages and limitations of PETA.

8.1 Evaluation Metrics

The ultimate goal of transcriptome assemblers is to reconstruct all expressed transcripts in full length. Obviously, the number of full length transcripts is a critical criteria for evaluation (106, 107, 108). In this study, we conclude that a reference transcript is assembled in full length if there is such an assembled transcript that covers at least 99% length of it, allowing at most 10 mismatches and indel base pairs. We set the criteria stringent because we find that some transcripts are highly similar. Theoretically, an optimal assembler is able to get all transcripts in the Oracle set. But in practice, some repeat segments or highly similar regions are difficult to go through.

In addition, we adopt the evaluation metrics suggested by (6) to evaluate the performance of the assembled transcriptome, given the Oracle set as reference transcripts, which has been introduced in the section above. The aim is to reconstruct all full length transcripts in the Oracle set.

The alignment is performed by Blat (78), which provides a good trade-off between efficiency and accuracy. We run Blat with default parameters. The command is like: "*blat oracle_set.fa contigs.fa -ooc=11.ooc contigs.oracle.psl*".

8.1.1 Accuracy

The accuracy metric is defined as the percentage of the correctly assembled bases estimated using the set of reference transcripts (N). It indicates how accurate the assembler is. Accuracy can be formally written as:

$$Accuracy = 100 * \frac{\sum_{i=1}^{M} A_i}{\sum_{i=1}^{M} L_i} \tag{8.1}$$

where L_i is the length of the alignment between a reference transcript and an assembled transcript T_i, A_i is the number of correct bases in transcript T_i, and M represents the number of best alignments between assembled transcripts and reference.

Highly similar transcripts may result in misleading accuracy values. For example, transcripts SPBC29A3.12_T0 and SPAC24H6.07_T0 locate at chromosome 1 and chromosome 2 respectively, but they are 90% similar. So for two alignments on the same transcript, we keep the one with less mismatches only.

8.1.2 Completeness

The completeness metric is defined as the percentage of reference transcripts covered by all the assembled transcripts. The covered regions may not be continuous. It is written as:

$$Completeness = 100 * \frac{\sum_{i=1}^{N} I(C_i \geq \delta)}{N} \tag{8.2}$$

where N is the number of reference transcripts, C_i represents the percentage of bases of the ith reference transcript, i, that are covered by some assembled transcripts. The indicator function, I, gives a value as either 1 or 0. If C_i is greater than some user defined threshold δ, say, 80%, the indicator function I would give a value 1. This metric indicates how many reference transcripts are reconstructed with at least δ percentage, regardless how fragmented the assembled transcripts are.

8.1.3 Contiguity

The contiguity metric is defined as the percentage of expressed reference transcripts covered by a single, longest assembled transcript. It is similarly written as:

$$Completeness = 100 * \frac{\sum_{i=1}^{N} I'(C_i \geq \delta)}{N} \tag{8.3}$$

where the indicator function I' gives value 1 if there exists such a single, longest alignment which represents a percentage greater than some user defined threshold δ, say, 80%.

As a supplementary metric, we also derive N50 value of the longest alignments. For every reference transcript, we pick the longest continuous alignment into the list H. The size of H is then the same as the number of reference transcripts. The N50 value is calculated as:

$$Aligned\_N50 = N50(H) \tag{8.4}$$

Greater N50 value indicates better contiguity. For reference, the optimal N50 is also calculated based on the total length of all reference transcripts.

8.1.4 Chimerism

A chimeric transcript is an assembled transcript that contains non-repetitive fragments (at least 50bp) from two or more different reference genes. They can arise from biological sources (gene fusions or trans-splicing), experimental sources (intermolecular ligation) or informatics sources (misassemblies). Misassembled chimeric transcripts can be distinguished from true chimaeras by determining whether the number of reads spanning the chimeric junction is significant when compared to the number of reads spanning other segments of the transcript. Paired-end reads spanning the chimeric junction also help for distinguishment.

Table 8.1 summaries the metrics we are using.

Metric	**Description**	**Example**
Full length transcripts	99% covered, < 10bp indels and mismatches	2,000
Accuracy	How accurate of the assembled bases	95.2%
Completeness 80%	Percentage of reference transcripts with > 80% covered	98.6%
Contiguity 80%	Percentage of reference transcripts with > 80% of continuous region covered	94.3%
Aligned N50	N50 of longest continuous alignments	1,700
Chimerism	Percentage of assembled transcripts aligned to different genes	4.3%

Table 8.1: Evaluation Metrics

The tail length (k) for running PETA is set to 25 (read length is 68bp). The parameters we used to run Trinity were ”–SS_lib_type FR –CPU 8 –min_contig_length 100”. Other assemblers are run with the default parameters.

We develop our evaluation module in Python, which is also available on the homepage. The transcripts reported by the assemblers are aligned to annotated transcripts in Blat with default parameters. For the accuracy metric, we count the correct bases obtained by the assemblers. However, at a single locus, an assembler may report two transcripts but with different nucleotides due to sequencing errors or SNPs. So we take the longest continuous alignment for consideration only. A longer transcript is supposed to be more reliable. If the bases on a long transcript is not correct, its performance is worse.

A transcript is categorized as full-length if there an alignment on a transcript such that:

- The similarity is larger than 99%.
- The alignment covers 99% of the transcript.
- There is no more than 1 indel on both query and reference transcripts.
- The number of mismatches and indel bases are no more than 10bp.

8.2 Results of *S.pombe* Dataset

The evaluation results of the *S.pombe* dataset are visualized in the Figure 8.1 and Figure 8.2. We can conclude that PETA outperforms other assemblers in terms of full-length transcripts number, aligned N50 and contiguity 80%. Only another *de novo* assembler IDBA-Tran is comparable with PETA. And surprisingly, the reference based assembler Cufflinks shows a bad performance. That is because there are not many splicing events in the *S.pombe* dataset, and Cufflinks does not merge disjointed transcripts if they don't have enough overlapping length. The results suggest that *de novo* transcriptome assemblers have their advantages given simpler transcriptomes.

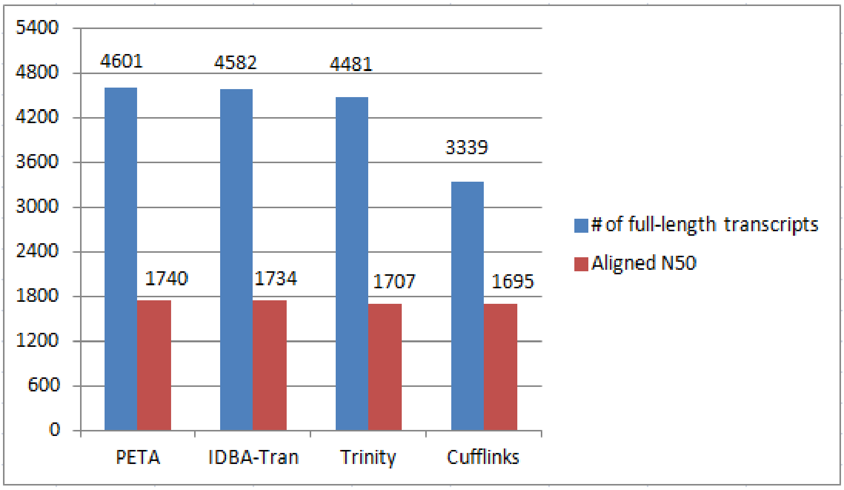


Figure 8.1: Number of full-length and Aligned N50 of *S.pombe* -

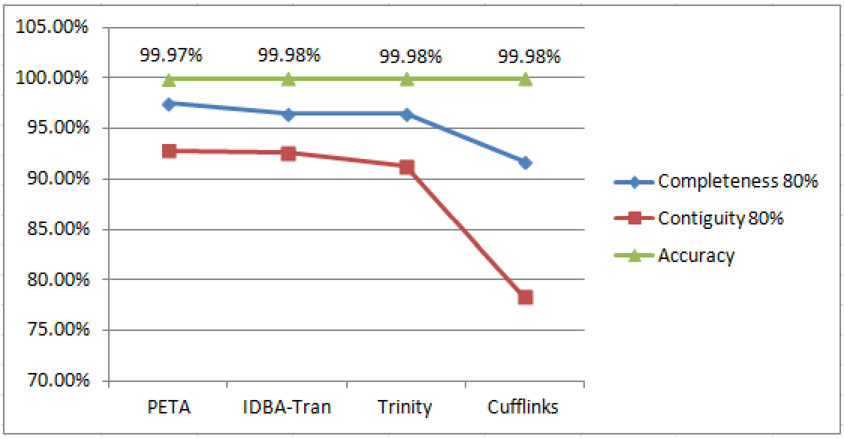


Figure 8.2: Accuracy, Completeness 80% and Contiguity 80% of *S.pombe* -

We intersect the full-length transcripts obtained by PETA, IDBA-Tran and Trinity. Although the read coverage is on average as high as more than 30X, the set of transcripts are diverse to some extend. In next section, we dive into the implementation details to investigate the reasons why PETA fails to assemble some transcripts.

8.3 Results of Human Dataset

We also run PETA, IDBA-Tran, Trinity and Cufflinks on the human RNA-seq dataset SRX011545. Since the human transcriptome is much more complex, the performance of Cufflinks is much better than other three *de novo* assemblers. The comparison can be found in Figure 8.4 and 8.5.

PETA obtained fewer full-length transcripts than IDBA-Tran, but is much better than Trinity. The results suggest that PETA performs well even if the transcriptome is complex. But since the template connection of PETA relies on the paired-end reads locally, errors may be introduced to PETA results.

8.4 Evaluation on Dataset with Lower Coverage

Higher coverage is a big advantage of RNA-seq. Recently, single cell RNA-seq is becoming more popular in studying the transcriptomes at different time frames. The median read coverage across expressed transcripts is 53.8% in the Quartz-Seq

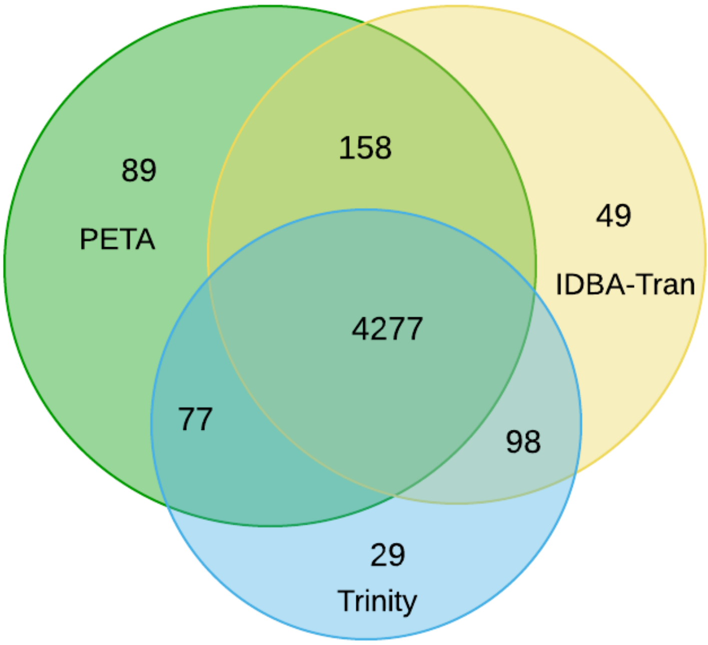


Figure 8.3: Intersection among PETA, IDBA-Tran and Trinity for *S.pombe* -

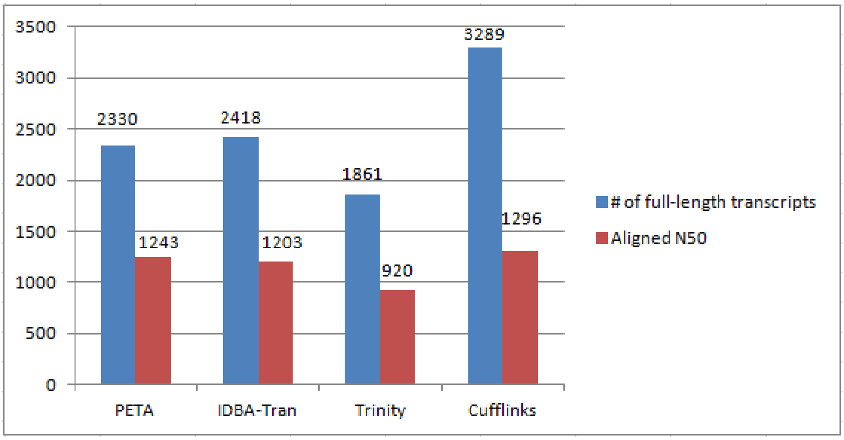


Figure 8.4: Number of full-length and Aligned N50 of human dataset -

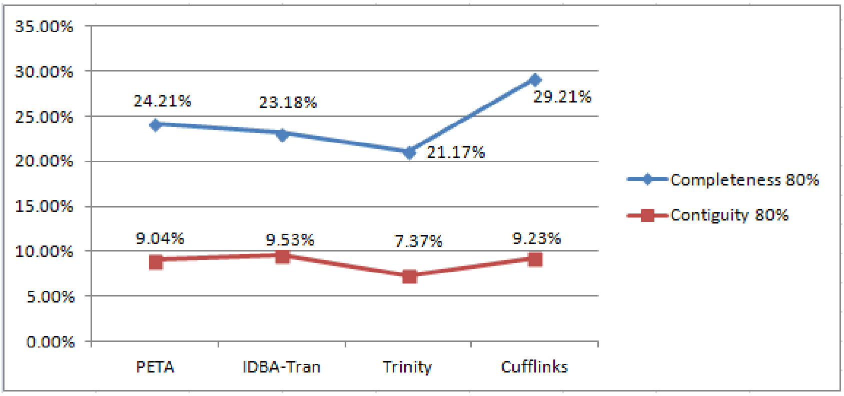


Figure 8.5: Accuracy, Completeness 80% and Contiguity 80% of human dataset -

method, compared with 84.4% in conventional RNA sequencing (142). We claim that PETA performs even better for low coverage datasets.

In order to benchmark the performance of the assemblers given a dataset with lower coverage, we select a subset of the *S.pombe* dataset. The accession id is SRR097897 [1]. The size is one quarter of the full *S.pombe* dataset. There are 7.5 million paired-end reads with length 68bp.

The results are listed in Table 8.2. We run Cufflinks, Trinity, IDBA-Tran and PETA on the same dataset. The aim is to obtain as many full-length transcripts in the annotated reference.

Metric	Cufflinks	Oases	Trinity	IDBA-Tran	PETA
# of contigs	3,951	8,102	7,952	6,023	8,165
Full length	3,244	3,247	3,077	3,575	3,694
Aligned N50	1,682	1501	1,422	1,544	1,569
Accuracy	99.98%	99.93%	99.97%	99.97%	99.97%
Completeness 80%	91.98%	87.43%	85.47%	86.38%	88.42%
Contiguity 80%	78.26%	73.80%	70.83%	78.92%	80.85%
# of chimaeras	160	80	47	67	97
Chimerism	4.05%	1.14%	0.59%	1.11%	1.19%

Table 8.2: Experiment Results

[1] SRR097897: http://sra.dnanexus.com/runs/SRR097897

The alignment results are analyzed to obtained the results in Table 8.2. Figure 8.6 and 8.7 compares the performance in charts.

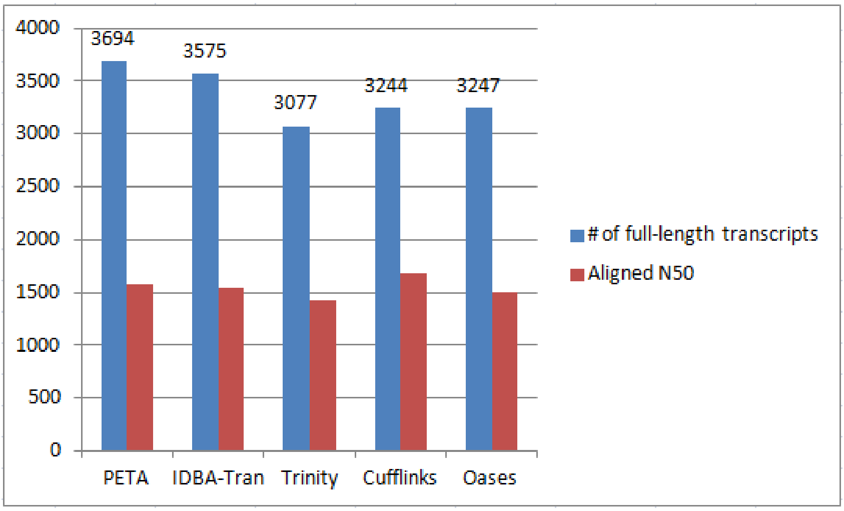


Figure 8.6: Number of full-length and Aligned N50 of SRR097897 -

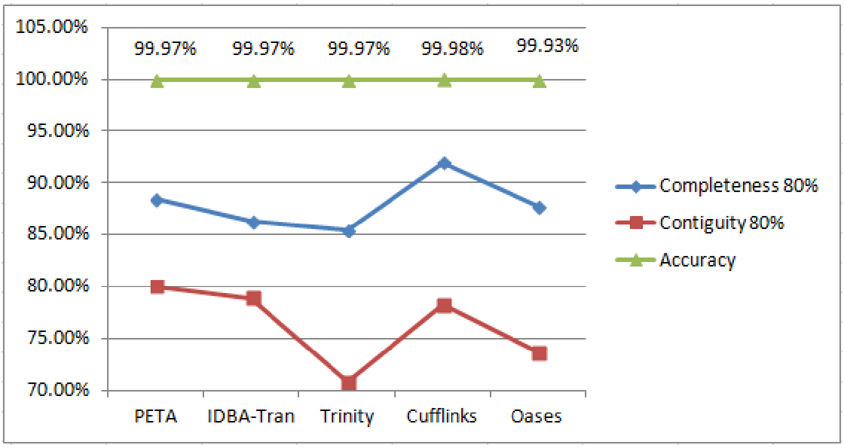


Figure 8.7: Accuracy, Completeness 80% and Contiguity 80% of SRR097897 -

From the results we can conclude that PETA obtains the most full-length transcripts and its contiguity 80% is the highest. It indicates that PETA is able to capture full-length transcripts while keeping the accuracy. Trinity achieves best Completeness 80%, however, it reports lower value of contiguity 80%, which indicates that Trinity has difficulty in resolving the alternative splicing. Out of all assemblers, Oases performs significantly worse than others. We suspect that the merging various contigs from different k-mer graphs are still challenging in practical implementations.

We also draw the Venn diagram among the full-length transcripts obtained by PETA, IDBA-Tran and Trinity. It is shown in Figure 8.8.

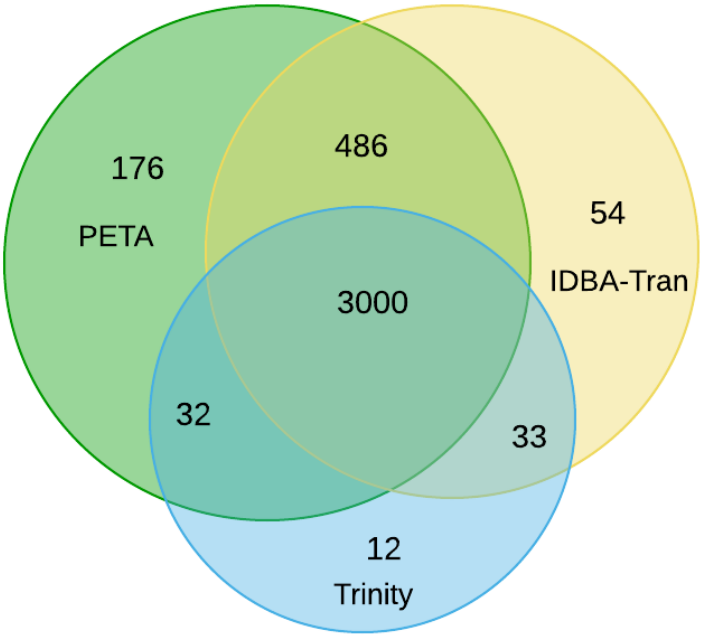


Figure 8.8: Intersection among PETA, IDBA-Tran and Trinity for SRR097897 - The numbers in the diagram are the numbers of full-length transcripts.

The intersection results indicates that even if PETA obtains most full-length transcripts, he still miss 99 transcripts obtained by IDBA-Tran and Trinity. Comparing the result of IDBA-Tran and Trinity, they also report a different set of transcripts.

Based on the observation above, we can prove that transcriptome is complex. Even for a transcriptome with a few alternative splicing, the complete set of expressed variants are hard to be reported. We discuss more about this set of results in Section 8.6.

To draw a conclusion, PETA performs the best in terms of most evaluation metrics. It reports accurate full-length transcripts efficiently, especially good at assembling low coverage transcripts.

8.5 PETA Browser

In order to investigate the assembly process in detailed, we have developed a visualization web application PETA Browser to visualize the assembled contigs and raw reads aligned to the annotated transcripts. It is a powerful tool to help researchers to find enough information about the assembly process. The main feature of the visualizer is to show alignments to the annotated transcripts.

An alignment is represented by a solid rectangle, which occupies one row on the webpage. The blue color in rectangle means that the reverse complement of the query is mapped. Otherwise, the alignment is in green color. Red bars at the head or tail indicate the soft-clipped portion of the alignment. For paired-end reads, if both of them are found, they are connected by a solid gray line. Otherwise, a red triangle is appended to the alignment. PETA Browser is able to show the mismatches (small red bar on the rectangle), insertion (a yellow bar on the rectangle) and deletion (a dashed line connecting fragmented portions).

Once the mouse is moved onto an alignment, a tooltip box will pop up to display detailed information of the alignment. It includes the name of the query, number of mismatches and insertion bases, the position of its mate read, etc. A vertical ruler is shown to help to locate interested regions. As the mouse moves, a number is updated to show the current locus on the transcripts. The users can also configure the plotting parameters such as defining the height of the alignment rows.

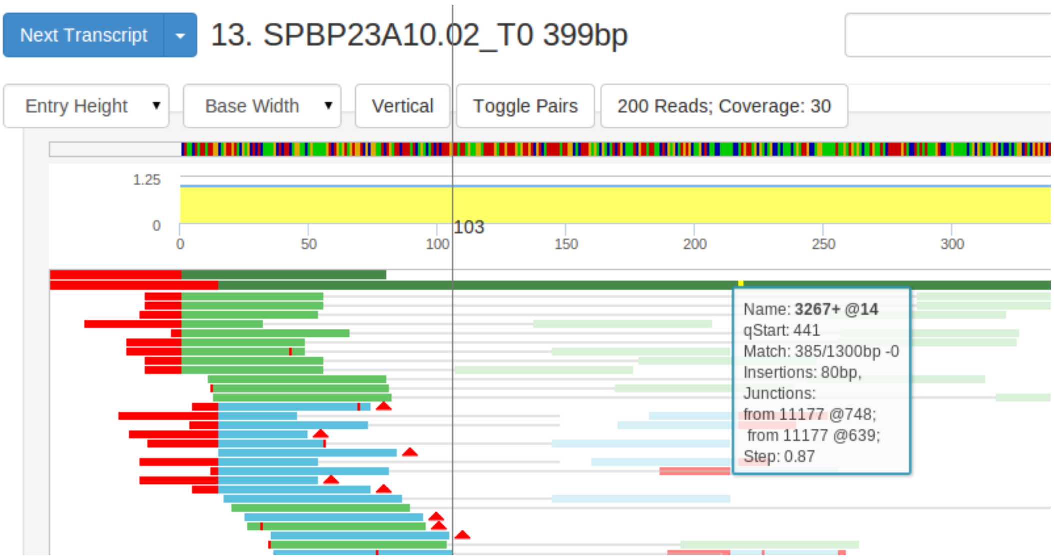
Figure 8.9 is a screen shot of PETA Browser.

PETA Browser is implemented in Python, using the web framework Django. And the real-time plotting is fulfilled by HighCharts.js [1], which draws high resolution SVG images on the webpage.

It accepts the standard PSL alignment files produced by the mapper Blat. It is easy to be configured. As long as the users provide a list of annotated transcripts and the alignment files, PETA Browser will handle all interaction requests.

The source code of PETA Browser is also provided as an open-source package at the homepage http://www.caishaojiang.com/peta.

[1] HighCharts.js: http://www.highcharts.com/



13. SPBP23A10.02_T0 399bp

200 Reads; Coverage: 30

Name: **3267+ @14**
qStart: 441
Match: 385/1300bp -0
Insertions: 80bp,
Junctions:
from 11177 @748;
from 11177 @639;
Step: 0.87

Figure 8.9: PETA Browser - For an annotated transcript *SPBP23A10.02_T0*, following information is collected to help the testing: (1). the yellow curve indicates whether there are repetitive 25-mer within the transcript. For example, the value at locus 100 is 1, meaning that the 25-mer starting at locus 100 appears only once within the transcript. (2). assembled contigs. PETA have reported two contigs, which are plotted right under the yellow curve. The color of contig alignments are darker and the height is greater. (3). raw reads. We map the reads to the annotated transcript and visualize them. (4). coverage is calculated.

8.6 Discussions

The experiments show that the performance of PETA is comparable with other assemblers. In this section, we analyze the different performance of PETA and IDBA-Tran in detailed.

From the Venn diagram in Figure 8.8, we can find that PETA reconstructs 208 full-length transcripts that are missed by IDBA-Tran. After investigations, we find that PETA assembles through the low coverage regions with the help of paired-end reads. It is the largest advantage of PETA.

However, it misses some full-length transcripts that are reported by IDBA-Tran. Here we investigate the 87 missing cases (from the *S.pombe* dataset SRR097897) using PETA Browser and analyze the limitations of PETA.

Figure 8.10 categorizes the missing cases into six types. In the following subsections, we analyze them case by case.

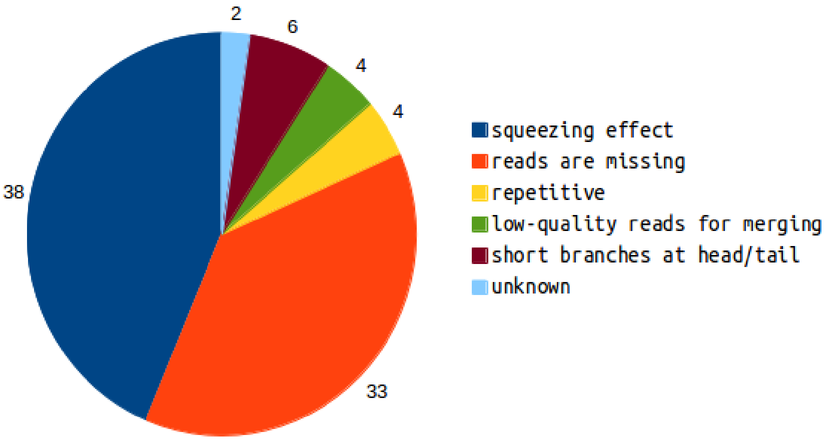


Figure 8.10: Reasons for missing full-length transcripts - Majority of the missing cases are because of *squeezing effect* and *missing reads*

8.6.1 Squeezing Effect

We define the *Squeezing effect* as the phenomenon that bad-quality reads are *squeezed* to form other noisy templates which are hard to distinguish in some cases.

Due to the sequencing errors, some reads are of low quality: too many mismatches, insertion/deletion on the reads and artefacts. Since we allow at most two mismatches, the low-quality reads would not be used for the extension. These reads are *squeezed* out during the extension. If there are multiple low-quality reads that are overlapped, later PETA may start extension from some of them and assemble another template, which is very noisy. We have observed many cases. Figure 8.11 illustrates this case.

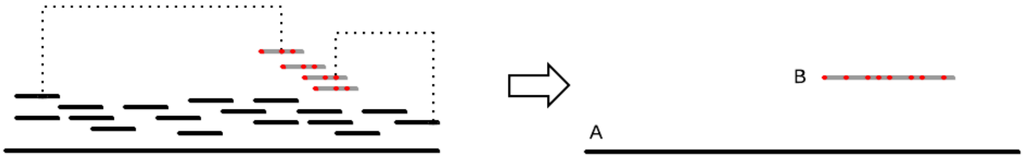


Figure 8.11: Squeezing effect - The solid black line at bottom represents a template, and shorter lines represent reads. The dashed lines connecting two reads indicate paired-end reads. The four gray reads with red dots represent the reads with too many mismatches. PETA will construct two templates A and B finally.

Squeezing effect is difficult to solve because the noisy templates may be also supported by paired-end reads, such as the example in Figure 8.11. In the connection stage, template A and B may be connected because of the paired-end support! Currently PETA performs similarity checking before the connection. However, we also observe that in some cases, two transcript variants from different location of the genome can be highly similar (>80%). In this case, it is difficult to distinguish the noisy templates from the valid templates.

Around 44% of the missing cases are caused by squeezing effect. We need to design more sophisticated approaches to deal with this case.

8.6.2 Reads are Missing

Around 40% of the missing cases are because that we do not utilize some reads. Ideally PETA should consume all of the raw reads in the dataset. However, if we try to start extension from every read, it is too time consuming to be acceptable. That is why we perform grouping of the reads to determine the starting reads. In this case, after assembly, actually there are still some reads which are useful but are not touched.

That may result in three kind of missing transcripts:

- The transcript expression level is very low. Only a few reads origin from it. All of the reads on the transcript are not used by PETA.
- Short exons (<100bp) are not captured because there are few reads at the junction. And these reads are not used.
- At the head/tail of a transcript, the overlapping length is too short and there are few reads at the head/tail. So the head/tail portion is missing. An example is shown in Figure 8.12.

For IDBA-Tran and Trinity, since they build the graph by exhaustively consuming all k-mers in the reads, they are able to fully utilize all k-mers.

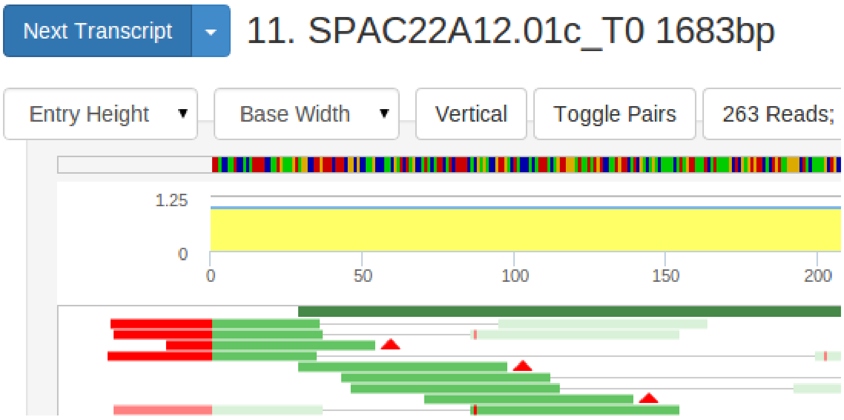


Figure 8.12: Reads are missing - Only one contig is reported by PETA. At the head of the transcript, the four reads are not utilized.

There is always trade-off between efficiency and accuracy. To capture such kind of missing transcripts, a better solution is that, after linear extension, identify high quality reads in the remaining unused reads and extend them.

8.6.3 Short Branches at Head/Tail

6 missing cases are caused by ambiguities at the head/tail area of the transcripts. Figure 8.13 illustrates an example.

PETA Performs connection between templates. However, to avoid the squeezing effects, we don't allow introducing blocks that are both short and at the head/tail of a template (refer to Section 7). In this case, we will pick the direction with more reads supporting the extension.

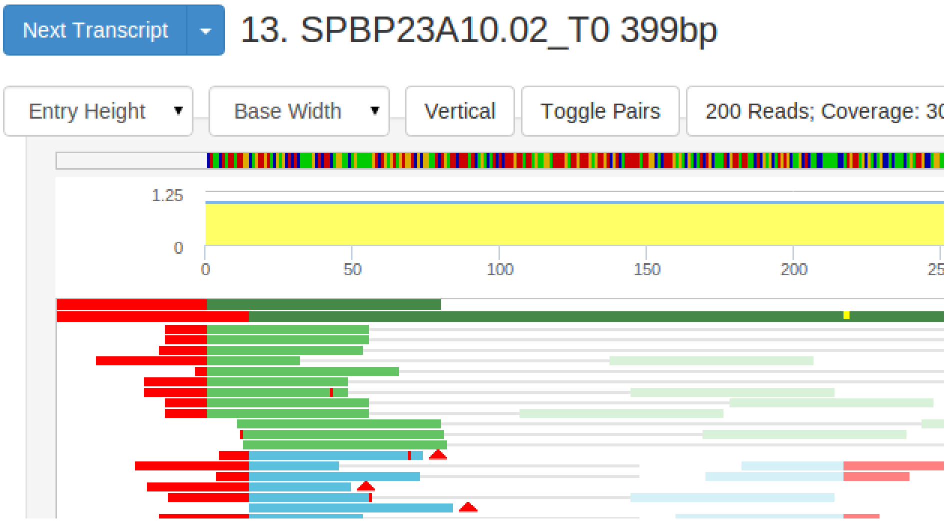


Figure 8.13: Ambiguities at head/tail - At the begging region of the transcript *SPBP23A10.02_T0*, there are two groups of reads (green and blue color) supporting different branches to extend.

Unfortunately, such cases seem not solvable by PETA.

8.6.4 Low-Quality Reads for Merging

This is a minor case for the missing transcripts. At a region where the overlapping length between reads are short, there is some read with insertion/deletion that covers the region. K-mer based IDBA-Tran can utilize it, but PETA cannot pick the read for extension, resulting two disjointed templates. Figure 8.14 is an example.

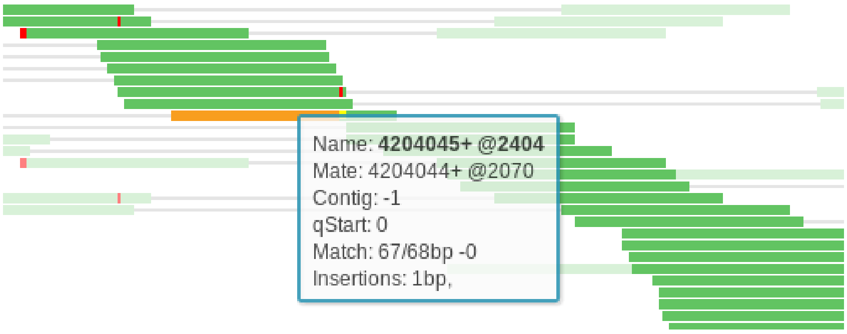

Name: **4204045+ @2404**
Mate: 4204044+ @2070
Contig: -1
qStart: 0
Match: 67/68bp -0
Insertions: 1bp,

Figure 8.14: Low-quality reads for merging - The read 4204045 has a 1bp insertion. PETA cannot make use of it for extension.

9

UASIS - Universal Automated SNP Identification System

9.1 Backgrounds

9.1.1 Heterogeneous Representations of SNPs

SNP, or Single Nucleotide Polymorphism, is defined as a bi-allele polymorphism at a single base with a frequency of more than 1% in the population (57, 58). Around 90% of the genome variations are limited to SNPs (60), which have been proven to be of great value for medical diagnostics and developing pharmaceutical products. They can also help identify multiple genes associated with complex diseases such as cancer and diabetes (143, 144, 145).

With the publication of the Human Genome Project (HGP) and emergence of next generation high-throughput sequencing techniques, there has been an explosion of data available for public use. SNP databases such as dbSNP (146), GWAS (formerly HGVbaseG2P) (147), HapMap (148) and JSNP (149) have collected millions of records. dbSNP, the largest one maintained by the National Center for Biotechnology Information, has collected 38,077,719 SNPs (rs#'s) for *Homo sapiens* to date (May 24, 2011, Build 132). The amount of data has been growing significantly. In addition, there are many more SNP databases, either public or private, that are used for pharmacogenetic research. An universal nomenclature is critical for clear, unequivocal and effective communication.

However, it is widely recognized that heterogeneity of SNP nomenclatures and notations has complicated the process (60, 150, 151, 152, 153). Table 9.1 lists the numerous alternative manners of designating a SNP in major databases. To make matter worse, private databases continue to use non-conventional representations that enlarge the set of possible nomenclatures as shown in Table 9.2 (60).

Database	**SNP Names**
dbSNP	rs3737965
	ss4923964, ss69366921
HGVBaseG2P	HGVM2256489
HGVS	NM_001286.2:c.87+45G>A, NM_021735.2:c.87+45G>A
	NM_021736.2:c.87+45G>A, NM_021737.2:c.87+45G>A
	NT_021937.19:g.7871183G>A
JSNP	IMS-JST083663
PharmGKB	rs3737965@chr1:11789038
HapMap	rs3737965

Table 9.1: Alternative Names of an SNP

There are many reasons for the existence of differing nomenclatures. Although Human Genome Variation Society (HGVS) has recommended widely-used guidelines for mutation notation, researchers of each laboratory have strong emotional attachment to their own naming system (154). Research articles that first report novel SNPs do not always follow the HGVS guidelines, and the final genomic sequence is complied over many separate entries. Previous nomenclatures sometimes subsist for historical reasons. For example, *rs28942082* is still recorded as *”FH NAPLES”* or *”Bly544Val”* in OMIM (see Table 9.2).

9.1.2 Problems of Current SNP Nomenclatures

Unambiguous and correct descriptions of SNPs in databases and in the literature are of utmost importance, not in the least since mistakes and uncertainties may lead to undesired errors in clinical diagnosis. HGVS nomenclature guidelines were proposed in as early as 1998 (155) then extended later on (156, 157). The guidelines

dbSNP	rs28942082
Genome-browser-like Syntax	Chr19:11,087,877-11,087,877 G/T
	Chr19:11087877 G/T
Others	geneA,11,EXON,108,T,hetero
	geneAsynonym,11,108,exon,GT
	proteinB, Gly564Val; proteinB, Bly544Val
	0014 FH NAPLES

Table 9.2: Alternative Names of an SNP

have since been improved regularly[1]. However, the sole existence of the guidelines by themselves is not sufficient. The standardization of SNP identification is far from complete (150, 151, 153).

It is clear that dbSNP is becoming a major center for deposition of SNPs from various sources. The SNP nomenclature of dbSNP, rs#, is unique, clear and stable. It has been widely adopted and heavily referenced in the literature. JSNP, GWAS, HapMap and PharmGKB provide corresponding rs# when displaying their own records. We highly respect its authority.

It is noted that overlapping of SNPs is very low (around 1%) among recognized databases (151). JSNP reported only 20.9% identity compared to dbSNP (149). Researchers have to submit their SNPs to dbSNP before they can get a rs#. However, some SNPs discovered in the research or diagnostic laboratory may even never be reported in any publication or database. Some SNPs have considerable delays in their public release due to commercial agreements, legal considerations or ethical reasons (145, 158). They are unlikely to be assigned identifiers that can be uniformly used later on. Even for dbSNP itself, there are many rs#'s abandoned due to regular clustering (159). These identifiers may have been cited in publications, leading to confusion and ambiguity.

Another candidate is HGVS mutation nomenclature guidelines, which are largely adopted by researchers and enforced by some journals. The format is like *"<Accession Number>.<version number>(<Gene symbol>):<sequence type>.<mutation>"*. However, it is not universally applied as a standard, since it is complex and not unique.

[1] http://www.hgvs.org/mutnomen/

Table 9.1 gives five alternative names that are legal for a SNP, where the coordinate systems are based on different reference sequences. The mutation position is obtained based on some reference sequences. In addition, reference sequences are evolving with each new version. That makes the names unstable. More effort is thus required to translate data in published papers and databases between different versions of reference sequences (160, 161). Finally, the names may be too long and complex to remember and communicate.

Current SNP nomenclatures, including rs#, are mostly arbitrary combination of letters and digits maintained by manual curation. The major problem is that they are not informative and only available within a single database. Automatic ways of mapping SNPs based on their names are rare. One way is to perform searching in available databases separately, and then compare the obtained records manually. For example, given only SNP names, we are unable to answer these kind of simple questions: *What SNPs have been discovered on gene CHR1 (chromosome 5, locus 26648951..26653073)?* or *What diseases have been found closely associated to rs28942082?* HGVS nomenclature is searchable and informative, but suffers from complexity and non-unique feature.

With differing nomenclatures, it is difficult to cross reference SNPs among the various databases. Research based on the data only from one SNP database will lead to an incomplete compilation of variants and inadequate genomic analysis. For researchers who track SNPs through literature scanning, it is very difficult to gain a global picture from overwhelming publications since SNPs are not uniformly searchable in the literature. It is also not possible to search by position or polymorphism information. That could be a tough data mining challenge, which consumes considerable resources and time. From the discussion above, we believe that the existing SNP nomenclatures do not provide a universal standard.

9.1.3 SNP Standardization and Database Integration

Tremendous efforts have been made to keep SNP data uniformly. Besides the continuous development of HGVS nomenclature guidelines, SNP databases are integrating data from more sources.

GWAS, previously HGVbaseG2P, is one of the largest SNP databases (162, 163). It gathers information of SNPs from the literature, their own and collaborative discovery efforts and unsolicited submissions. It exchanges core data with dbSNP regularly. The pharmacogenomics knowledge base (PharmGKB) allows cross-referencing against dbSNP, JSNP and HapMap, as well as other sources such as UCSC Genome Browser (164).

Some applications focus on retrieving SNPs fulfilling certain criteria such as locus and haplotype tagging. SNPper is web-based platform to search and export SNP records from dbSNP (165). TAMAL (Technology And Money Are Limiting) provides a query portal to latest versions of five SNP sources (HapMap, Perlegen, Affymetrix, dbSNP and the UCSC genome browser) (166). It helps to select SNPs that are likely involved in the genetic determination of human complex traits. LS-SNP annotates from dbSNP the coding of non-synonymous SNPs (nsSNPs) that will result in mutation in protein (167). Other works place emphasis on intragenic SNPs (168).

Among the previous works carried out, Mutalyzer sequence variation nomenclature checker (153) and SNP-Converter (60) are similar to the work described here. These two applications aim to support HGVS nomenclature guidelines. Mutalyzer checks if an SNP name follows the HGVS guidelines. Furthermore, it is capable of generating legal identifiers given the pivot features of a SNP. SNP-Converter converts whatever SNP names into HGVS names by exploring certain gene databases to determine the correct locus. It treats the integration process as a knowledge mining task. SNP-Converter is based on a complete SNP notation in XML format, acting as an ontology, to create a uniform semantic environment (60, 169).

9.2 Implementations: Universal SNP Nomenclature and UASIS

From the discussions above, it is clear that dbSNP is an important database that cannot be ignored by any application. However, it does take considerable effort to translate nomenclatures among the SNP databases. To overcome the shortcomings of rs# and HGVS nomenclatures, we propose a universal nomenclature and UASIS (Universal Automated SNP Identification System). We believe our nomenclature is a good complement to rs# and HGVS, acting as a bridge connecting various databases, including private and unpublished ones.

A system of nomenclature has to strike a compromise between the convenience and simplicity required for everyday use and the need for adequate definition of the concepts involved (170). In 2006, Human Variome Project Meeting gathered leading representatives to discuss key problems of human gene variation industry (152). The meeting gave 96 recommendations. Two of them regarding to "Nomenclatures and Standards" are:

4. Develop tools to accurately translate and search earlier nomenclature systems into successor systems.*

6. The most current genome build be unambiguously adapted as the reference sequence, and that a standard be developed for the submission of all variant data that includes both a genome coordinate as well as sufficient flanking sequence to map the variation independently.

UASIS is inspired from these two requirements. UASIS proposes a universal nomenclature for SNPs with the form *"<human genome version> . <chromosome number>:<locus>:<alleles>"*. Detailed specification is shown in Table 9.3. According to this specification, SNP *rs3737965* is represented as *HG19.1*:*11789038*:*G/A*, indicating a pair of alleles *"G"* and *"A"* at position 11789038 of chromosome 1, and the position is based on human reference genome version 19. Note that for indels, the polymorphism occurs *at* the position given. For example, *"1234insT"* means that *"T"* is placed at position 1234, and the original one, say, *"C"* is at position 1235.

Syntax	**Example**	**Description**
HG(*numeric version*)	HG19	Complete human reference genome
		by UCSC. '19' is version number
Chr number	1..22, X, Y	Chromosome numbers
Numeric	21898363	1-based position
Nucleotides	**A**, **C**, **G**, **T**, **N**	**N** for unclear nucleotide
/	G/A	Substitution: alleles are 'G' and 'A'
ins	insA	Insertion: 'A' is inserted
del	delT	Deletion: 'T' is deleted

Table 9.3: Universal SNP Nomenclature

Compared to HGVS guidelines, we fix the coordinate to be the whole human genome. And we give only one position without "_", since we consider only single bi-allele mutations. The first advantage is that it allows for succinct comparison using the accession numbers. The nomenclature is based on the human reference genome and not any *arbitrary* reference sequences, resulting in the generation of unique identifiers. All SNPs would be given the same prefix *"HG19"* currently. Secondly, it is unambiguous, informative and stable since the name consists of all necessary information to uniquely define an SNP. More importantly, UASIS nomenclature gives names that are searchable and comparable. It helps SNP tracking in the literature if universally adopted.

Another difference is the representation of mutations. HGVS guidelines use a ">" symbol to mean "changed to". Here we only list all possible alleles delimited by a "/". *"A/T"* means that the major allele could be either *"A"* or *"T"*. Normally the first is the one on the reference genome. This definition is for simplicity. Determining the frequency of alleles requires more effort in the laboratory. In different populations or laboratory testings the results could be non-identical. For SNPs which have more than two alleles, the ">" symbol will lose its clarity, leading to ambiguity. This syntax is also used by other browser viewers (60). But we would recommend that the leftmost allele should be the major allele.

The most important advantage of UASIS nomenclature is that, unlike rs#, it does not depend on any particular database. The naming process of an SNP can be done automatically, regardless of the database maintaining it, or the contig the SNP is derived from, etc. Researchers do not necessarily submit to a particular database to get identifiers. They will get names instantaneously without waiting for manual approval using UASIS. Although dbSNP designates a ss# once a SNP is submitted, the ss# suffers similar problems of rs#. For private SNPs that cannot be published due to various reasons, UASIS nomenclature is obviously a better choice.

UASIS nomenclature is not intended to replace the rs# since rs# already has significant influence on SNP nomenclatures. rs#'s are simple, unique and stable. Actually, UASIS nomenclature is a good complementary to rs#, playing a similar role as ss#. But we believe that it is more than ss# and it will benefit the whole process of SNP standardization. One disadvantage of our notation is that it depends on the human reference genome. That is an unavoidable trade off given all attractive benefits of our universal nomenclature. But HG19 is considered as *"finished"* by the Genome Reference Consortium. We expect a much lower updating frequency of human genome in future.

UASIS is a web-based server system (http://www.uasis.tk) for annotating novel SNPs and cross-referencing among databases instantaneously. There are utility tools available, i.e., UASIS Aligner and Universal SNP Name Generator. For newly discovered SNPs, UASIS aligner performs efficient sequence alignment and checks whether the polymorphism has been deposited in main databases, including GWAS, dbSNP, JSNP and HapMap. In addition, for each mutation, UASIS provides an identifier based on our proposed nomenclature as described above. These identifiers can be used immediately and instantaneously. In this way, researchers are free to map SNPs among various nomenclatures. More databases like PharmGKB are currently in the process of being integrated into UASIS. Universal SNP Name Generator and SNP Name Mapper take in information of a SNP and perform cross-checking among main databases.

UASIS is available at http://www.uasis.tk since August 2010. It is implemented in PHP and MySQL, and designed for various types of web browser. Detailed information on the use of UASIS is provided online at the website.

9.2.1 UASIS Aligner

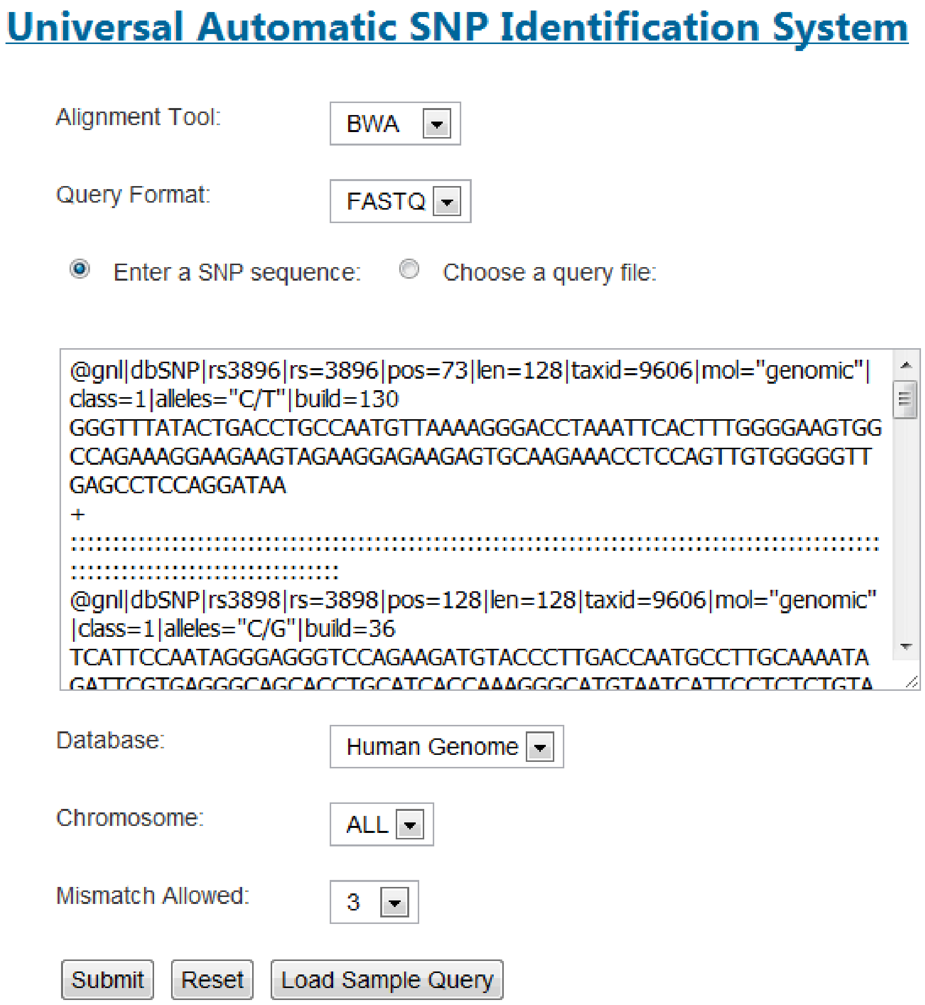
9.2.1.1 Input

Users upload flanking sequences of SNPs explicitly or by uploading a file in FASTQ format. They could choose underlying alignment tool, which chromosome to align, and how many mismatches allowed according to query characteristics. The human reference genome used is based on HG19, downloaded from UCSC[1]. Figure 9.1 showes the screenshot using the sample data.

9.2.1.2 Sequence Alignment

Efficiency and accuracy are critical for real time systems like UASIS. Bowtie (171) and BWA (77) are winners (159). They are able to align thousands of sequences every second. Both tools are developed based on Burrows-Wheeler Transform (BWT) (172) data structure and FM-index (173). Bowtie is customized for short reads

[1] http://genome.ucsc.edu/



Universal Automatic SNP Identification System

Alignment Tool: BWA

Query Format: FASTQ

● Enter a SNP sequence: ○ Choose a query file:

```
@gnl|dbSNP|rs3896|rs=3896|pos=73|len=128|taxid=9606|mol="genomic"|
class=1|alleles="C/T"|build=130
GGGTTTATACTGACCTGCCAATGTTAAAAGGGACCTAAATTCACTTTGGGGAAGTGG
CCAGAAAGGAAGAAGTAGAAGGAGAAGAGTGCAAGAAACCTCCAGTTGTGGGGGTT
GAGCCTCCAGGATAA
+
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::
@gnl|dbSNP|rs3898|rs=3898|pos=128|len=128|taxid=9606|mol="genomic"
|class=1|alleles="C/G"|build=36
TCATTCCAATAGGGAGGGTCCAGAAGATGTACCCTTGACCAATGCCTTGCAAAATA
GATTCGTGAGGGCAGCACCTGCATCACCAAAGGGCATGTAATCATTCCTCTCTGTA
```

Database: Human Genome

Chromosome: ALL

Mismatch Allowed: 3

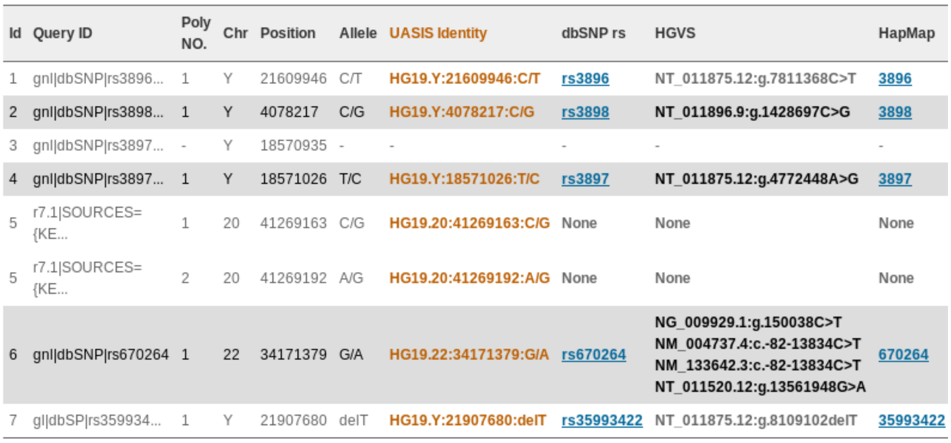
Figure 9.1: Input of UASIS Aligner - Users could choose to upload the flanking sequences of SNPs as file, or input the sequences directly. Currently we support FASTQ format only. Other parameters include which chromosome to align and how many mismatches allowed.

around 35 base pair. It supports up to 3 mismatches by enumerating all possible permutations. This strategy makes it ultra fast, but it does not support gapped alignment. BWA employs roughly the same idea but it implements gapped alignment.

Query sequences are uploaded and aligned to reference human genome by executing Bowtie or BWA. Then UASIS checks whether the query SNP exists in dbSNP, GWAS, JSNP or HapMap by inspecting the allele position. UASIS is very responsive since the alignment tools are efficient.

9.2.1.3 Output

Alignments will be listed in tabular form, including query id, allele position, alleles, UASIS identifier, dbSNP rs#, GWAS id, JSNP id, HapMap id. Given the polymorphism position, we are able to obtain corresponding identifiers recorded in dbSNP, JSNP and HapMap. If no record is found in a database, a *"none"* message will be displayed for that database. Results in SAM format can be downloaded for further analysis. Figure 9.2 illustrates the sample output of UASIS Aligner.



Id	Query ID	Poly NO.	Chr	Position	Allele	UASIS Identity	dbSNP rs	HGVS	HapMap
1	gnl|dbSNP|rs3896...	1	Y	21609946	C/T	HG19.Y:21609946:C/T	rs3896	NT_011875.12:g.7811368C>T	3896
2	gnl|dbSNP|rs3898...	1	Y	4078217	C/G	HG19.Y:4078217:C/G	rs3898	NT_011896.9:g.1428697C>G	3898
3	gnl|dbSNP|rs3897...	-	Y	18570935	-	-	-	-	-
4	gnl|dbSNP|rs3897...	1	Y	18571026	T/C	HG19.Y:18571026:T/C	rs3897	NT_011875.12:g.4772448A>G	3897
5	r7.1|SOURCES={KE...	1	20	41269163	C/G	HG19.20:41269163:C/G	None	None	None
5	r7.1|SOURCES={KE...	2	20	41269192	A/G	HG19.20:41269192:A/G	None	None	None
6	gnl|dbSNP|rs670264	1	22	34171379	G/A	HG19.22:34171379:G/A	rs670264	NG_009929.1:g.150038C>T NM_004737.4:c.-82-13834C>T NM_133642.3:c.-82-13834C>T NT_011520.12:g.13561948G>A	670264
7	gl|dbSP|rs359934...	1	Y	21907680	delT	HG19.Y:21907680:delT	rs35993422	NT_011875.12:g.8109102delT	35993422

Figure 9.2: Result of UASIS Aligner - Align the flanking sequences of SNPs submited by users. The alignment is performed by Bowtie or BWA. If the SNP is found, search it in databases dbSNP, JSNP, GWAS and HapMap.

9.2.2 Experiments

To evaluate the accuracy and efficiency of UASIS, we conducted experiments on simulated and real SNPs with length 35, 76, 128 and 512bp, and performed cross-checking between dbSNP and JSNP. CPU time on a quad core of a 2.4 GHz Xeon E5620 processor with 16G RAM and accuracy in percentage are evaluated (see Table 4).

94771 reads were simulated from the human genome (Build 37.1) using MetaSim (174) package following the error pattern of Sanger reads. Meanwhile, 72241 flanking sequences were downloaded from public databases dbSNP[1] and JSNP[2]. For Bowtie, we use the options "–best -k 2 -v 3", meaning that it will report at most two hits allowing three mismatches in decreasing quality order. And for BWA, the options are "-n 3 -o 3", meaning that the edit distance is at most three and there are at most three gaps.

For both dataset, all three tools were found to show reliability. As the read length grows, the accuracy improves. Bowtie generated higher error rate since it does not support gapped alignment. But Bowtie was very efficient, taking less than 4 seconds to process.

UASIS is also introduced briefly on CBAS-SYMBIO[3] 2010 held in Singapore. Approximately 30 people outside UASIS group have tested it.

9.3 Universal SNP Name Generator

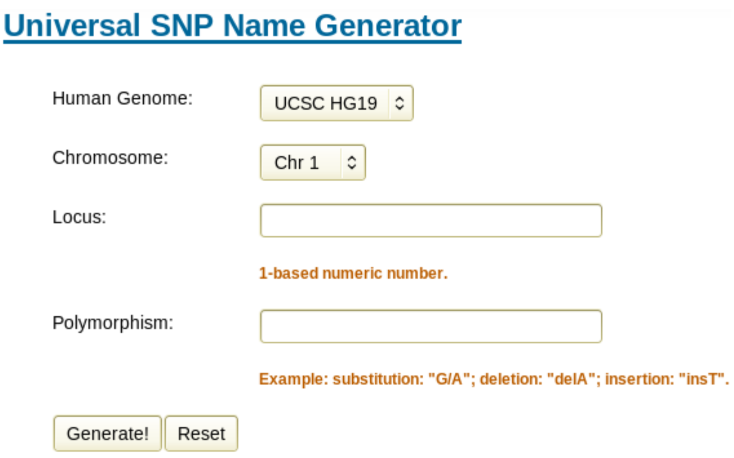
Similar to Mutalyzer (153), our generator takes in all pivot features that define a SNP uniquely. The features include reference genome, chromosome, position and alleles. Please note that for the mutation position of SNPs, different databases use different coordinate. dbSNP, the largest public one, uses 1-based positions. However, in the dump database files, the position is 0-based. And JSNP uses 1-bases positions in its dump database file. Here we choose 1-based strategy for consistency. The generator performs validation strictly to ensure the user input is legal. Figure 9.3 is a screenshot of the input page.

But instead of HGVS names, we generate our UASIS identifiers as the result, as well as corresponding HGVS names and access ids in dbSNP, GWAS, HapMap and JSNP. Currently GWAS is not providing downloadable SNP files, so we utilize the online query system of GWAS with rs# as the keyword. HGVS and JSNP

[1] ftp://ftp.ncbi.nih.gov/snp/organisms/human_9606/rs_fasta/

[2] http://snp.ims.u-tokyo.ac.jp/map/Dump/

[3] http://symbio2010.rsg.sg/



Universal SNP Name Generator

Human Genome: UCSC HG19

Chromosome: Chr 1

Locus:

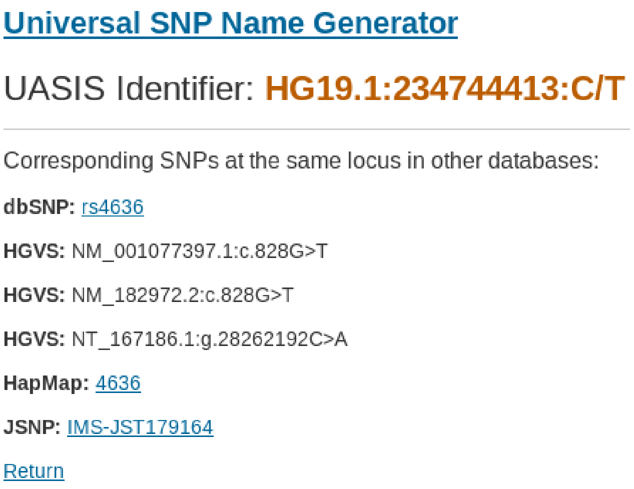
1-based numeric number.

Polymorphism:

Example: substitution: "G/A"; deletion: "delA"; insertion: "insT".

Figure 9.3: Input of Universal SNP Name Generator - Show the input options of Universal SNP Name Generator. Users are supposed to provide human genome version, chromosome number, locus and alleles.

identifiers are obtained from local databases recording relationship between them. When performing the cross-referencing, we only check whether there is a SNP at the same locus, regardless the alleles. But it is now sufficient for researchers. More functionality is under development. Figure 9.4 is the output of sample data.



Universal SNP Name Generator

UASIS Identifier: HG19.1:234744413:C/T

Corresponding SNPs at the same locus in other databases:

dbSNP: rs4636

HGVS: NM_001077397.1:c.828G>T

HGVS: NM_182972.2:c.828G>T

HGVS: NT_167186.1:g.28262192C>A

HapMap: 4636

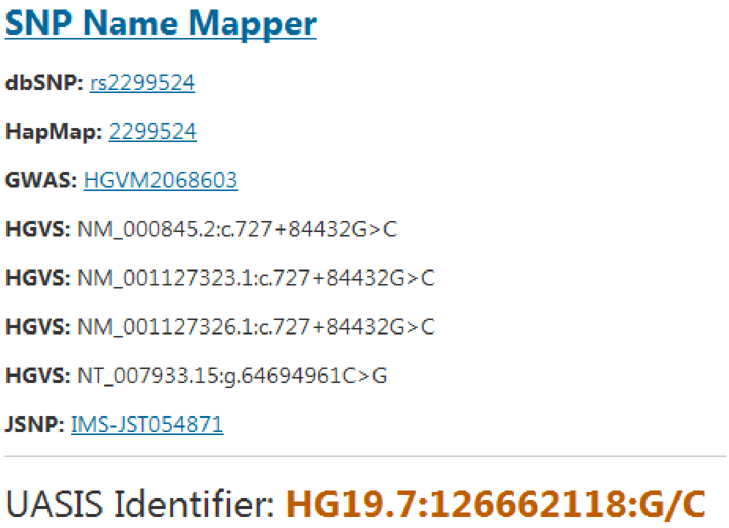
JSNP: IMS-JST179164

Figure 9.4: Result of Universal SNP Name Generator - Generate UASIS identifier given the pivot features. If there are records deposited in existing databases, show corresponding identifiers and links.

9.4 SNP Name Mapper

The SNP Name Mapper performs similar task to Name Generator. However, it is more suitable for researchers who have some SNPs at hand, and would like to know what related works have been done in the literature. Users are required to provide an existing SNP name from certain database. For example, *"rs3897"* from dbSNP. If the input name is not valid, a *"None"* message will be displayed.

Figure 9.5 illustrates a sample output of this utility. We also generate corresponding identifier following our universal nomenclature (see Section Implementation). The alleles information can only be obtained from two sources. If a JSNP record exists, there is alleles deposited. Otherwise, we search the online query system of dbSNP and parse the result page to extract the alleles information. If no rs# is available, we would not generate UASIS identifier.



SNP Name Mapper

dbSNP: rs2299524

HapMap: 2299524

GWAS: HGVM2068603

HGVS: NM_000845.2:c.727+84432G>C

HGVS: NM_001127323.1:c.727+84432G>C

HGVS: NM_001127326.1:c.727+84432G>C

HGVS: NT_007933.15:g.64694961C>G

JSNP: IMS-JST054871

UASIS Identifier: **HG19.7:126662118:G/C**

Figure 9.5: Result of SNP Name Mapper - Generate UASIS identifier given a particular SNP name. If there are records deposited in existing databases, show corresponding identifiers and links.

9.5 Availability and Requirements

Project name: UASIS (Universal Automated SNP Identification System)

Project home page: http://www.uasis.tk with no requirement of log-in

Operating system(s): e.g. Platform independent

Programming language: C++ and PHP web interface

10

Conclusion

In this dissertation, we discuss two studies based on Next Generation Sequencing (NGS) data. They are PETA (Paired End Transcriptome Assembler) and UASIS (Universal Automated SNP Identification System) respectively.

NGS RNA-seq technologies have started to reveal the complex landscape and dynamics of the transcriptome in an unprecedented level of sensitivity and accuracy. It has been applied to successfully capture transcriptome from yeast to human. Characteristics of RNA-seq data pose great challenges for accurate transcriptome assemly, especially for species without a high-quality reference genome.

Current *de novo* transcriptome assemblers are mostly based on *de Bruijn* graph, which has inherited problems in dealing with sequencing errors and low-expressed genes. Paired-end information is lost when constructing the graph. This important information is only used for post-processing.

In this study, we implement a new *de novo* transcriptome assembler PETA (Paired End Transcriptome Assembler), which weights heavily on paired-end information of RNA-seq libraries. We return to the overlap-layout-consensus approach. Paired-end information is used for contig extension, merging and validation.

PETA first creates a hashtable for the RNA-seq library to speed up the pairwise alignment. Then all reads are grouped to find frequent ones as the starting point of assembly. With the help of paired-end reads, we are able to recover the lowly expressed transcripts. The resulting graph structure is much cleaner comparing with the *de Buijn* graphs constructed from k-mers. We developed sophisticated graph processing algorithms. Finally, we apply a powerful statistical model to optimize the read distribution among the paths, such that we are able to pick correct paths as our final transcripts.

Our experiments showed that PETA outperforms other start-of-art assemblers, including Trinity, Oases, Cufflinks and IDBA-Tran in terms of number of full-length transcripts, aligned N50, accuracy, completeness, contiguity and chimerism. Our implementation is efficient and scalable. We believe PETA performs well for large-scale RNA-seq libraries. It helps to reveal the complex expression of transcriptome.

Compared with PETA, a powerful software to assemble RNA-seq data without reference genome, UASIS focuses on data management of SNPs resulted by overwhelming NGS data. Differing SNP nomenclatures have been a large concern for a long period. UASIS (Universal Automated SNP Identification System) proposes an informative, unique and unambiguous nomenclature that serves as a good complement to the present methods of identifying SNPs. The universal nomenclature is important for naming newly discovered or unpublished SNPs. The most significant advantage is that it provides a bridge to cross reference SNP identifiers among various databases. UASIS is a platform to perform pairwise sequence alignment and cross referencing in real time (<20s). Currently SNPs from dbSNP, GWAS, JSNP and HapMap can be mapped to one another. More databases are being integrated into UASIS. UASIS not only helps to achieve uniform notation of SNPs in the literature, but also aid in determining accurate SNP genotypes and haplotypes.

This thesis contributes to the bioinformatics community by providing two powerful tools for efficient processing and management of NGS data, especially for transcriptomics studies and related fields like GWAS.

References

[1] International Human Genome Sequencing Consortium. **Initial sequencing and analysis of the human genome**. *Nature*, **409**(6822):860–921, February 2001. 1

[2] Eric S. Lander. **Initial impact of the sequencing of the human genome**. *Nature*, **470**(7333):187–197, February 2011. 1

[3] Valerio Costa, Marianna Aprile, Roberta Esposito, and Alfredo Ciccodicola. **RNA-Seq and human complex diseases: recent accomplishments and future perspectives**. *European Journal of Human Genetics*, **21**(2):134–142, June 2012. 1

[4] Matthew L. Freedman, Alvaro N. A. Monteiro, Simon A. Gayther, et al. **Principles for the post-GWAS functional characterization of cancer risk loci**. *Nat Genet*, **43**(6):513–518, June 2011. 1

[5] Victor E Velculescu, Lin Zhang, Wei Zhou, et al. % bf Characterization of the Yeast Transcriptome. *Cell*, **88**(2):243 – 251, 1997. 1

[6] Jeffrey A. Martin and Zhong Wang. **Next-generation transcriptome assembly.** *Nature reviews. Genetics*, **12**(10):671–682, October 2011. 2, 23, 27, 35, 67, 68

[7] Zhong Wang, Mark Gerstein, and Michael Snyder. **RNA-Seq: a revolutionary tool for transcriptomics.** *Nature reviews. Genetics*, **10**(1):57–63, January 2009. 2, 19, 21

[8] Martin C. Frith, Michael Pheasant, and John S. Mattick. **Genomics: The amazing complexity of the human transcriptome**. *European Journal of Human Genetics*, **13**(8):894–897, June 2005. 2, 3

[9] Cole Trapnell, Brian A. Williams, Geo Pertea, et al. **Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation**. *Nature Biotechnology*, **28**(5):511–515, May 2010. 2, 27, 31

[10] Bjorn Schwanhausser, Dorothea Busse, Na Li, et al. **Global quantification of mammalian gene expression control**. *Nature*, **473**(7347):337–342, May 2011. 2

[11] Eric T. Wang, Rickard Sandberg, Shujun Luo, et al. **Alternative isoform regulation in human tissue transcriptomes.** *Nature*, **456**(7221):470–476, November 2008. 3

[12] Kishore R. Sakharkar Meena Kishore Sakharkar, Bagavathi S. Perumal and Pandjassarame Kangueane. **An Analysis on Gene Architecture in Human and Mouse Genomes**. *In Silico Biology*, **5**:347–365, 2005. 3

[13] Daisuke Hattori, Yi Chen, Benjamin J. Matthews, et al. **Robust discrimination between self and non-self neurites requires thousands of Dscam1 isoforms**. *Nature*, **461**(7264):644–648, October 2009. 4

[14] RIKEN Genome Exploration Research Group, Genome Science Group Genome Network Project Core Group, the FANTOM Consortium, et al. **Antisense Transcription in the Mammalian Transcriptome**. *Science*, **309**(5740):1564–1566, September 2005. 4

[15] Piero Carninci. **RNA Dust: Where are the Genes?** *DNA Research*, **17**(3):209, June 2010. 4

[16] M. D. Adams, J. M. Kelley, J. D. Gocayne, et al. **Complementary DNA sequencing: expressed sequence tags and human genome project.** *Science (New York, N.Y.)*, **252**(5013):1651–1656, June 1991. 4

[17] Kayoko Yamada, Jun Lim, Joseph M. Dale, et al. **Empirical Analysis of Transcriptional Activity in the Arabidopsis Genome**. *Science*, **302**(5646):842–846, October 2003. 4

[18] Jill Cheng, Philipp Kapranov, Jorg Drenkow, et al. **Transcriptional maps of 10 human chromosomes at 5-nucleotide resolution.** *Science (New York, N.Y.)*, **308**(5725):1149–1154, May 2005. 4

[19] Lior David, Wolfgang Huber, Marina Granovskaia, et al. **A high-resolution map of transcription in the yeast genome**. *Proceedings of the National Academy of Sciences*, **103**(14):5320–5325, April 2006. 4

[20] T. A. Clark, C. W. Sugnet, and M. Ares. **Genomewide analysis of mRNA processing in yeast using splicing-specific microarrays.** *Science*, **296**(5569):907–910, May 2002. 4

[21] Thomas E E. Royce, Joel S S. Rozowsky, and Mark B B. Gerstein. **Toward a universal microarray: prediction of gene expression through nearest-neighbor probe sequence identification.** *Nucleic Acids Res*, August 2007. 4

[22] M. S. Boguski, C. M. Tolstoshev, and D. E. Bassett. **Gene discovery in dbEST**. *Science*, **265**(5181):1993–1994, September 1994. 5

[23] D. S. Gerhard, L. Wagner, E. A. Feingold, et al. **The status, quality, and expansion of the NIH full-length cDNA project: the Mammalian Gene Collection (MGC).** *Genome Res*, **14**(10B):2121–2127, October 2004. 5

[24] V. E. Velculescu, L. Zhang, B. Vogelstein, et al. **Serial analysis of gene expression.** *Science (New York, N.Y.)*, **270**(5235):484–487, October 1995. 5

[25] Matthias Harbers and Piero Carninci. **Tag-based approaches for transcriptome research and genome annotation**. *Nature Methods*, **2**(7):495–502, July 2005. 5

[26] Toshiyuki Shiraki, Shinji Kondo, Shintaro Katayama, et al. **Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage.** *Proceedings of the National Academy of Sciences of the United States of America*, **100**(26):15776–15781, December 2003. 5

[27] S. Brenner, M. Johnson, J. Bridgham, et al. **Gene expression analysis by massively parallel signature sequencing (MPSS) on microbead arrays.** *Nature biotechnology*, **18**(6):630–634, June 2000. 5

[28] F. Sanger, S. Nicklen, and A. R. Coulson. **DNA sequencing with chain-terminating inhibitors.** *Proceedings of the National Academy of Sciences of the United States of America*, **74**(12):5463–5467, December 1977. 5

[29] Human Genome Sequencing ConsortiumInternational. **Finishing the euchromatic sequence of the human genome**. *Nature*, **431**(7011):931–945, October 2004. 5, 17

[30] Claire M. Fraser, Jeannine D. Gocayne, Owen White, et al. **The Minimal Gene Complement of Mycoplasma genitalium**. *Science*, **270**(5235):397–404, October 1995. 5

[31] David A. Wheeler, Maithreyan Srinivasan, Michael Egholm, et al. **The complete genome of an individual by massively parallel DNA sequencing**. *Nature*, **452**(7189):872–876, April 2008. 5

[32] David R. Bentley, Shankar Balasubramanian, Harold P. Swerdlow, et al. **Accurate whole human genome sequencing using reversible terminator chemistry**. *Nature*, **456**(7218):53–59, November 2008. 5

[33] Jay Shendure, Gregory J. Porreca, Nikos B. Reppas, et al. **Accurate multiplex polony sequencing of an evolved bacterial genome.** *Science (New York, N.Y.)*, **309**(5741):1728–1732, September 2005. 6

[34] Jay Shendure and Hanlee Ji. **Next-generation DNA sequencing**. *Nature Biotechnology*, **26**(10):1135–1145, October 2008. 6, 8, 11, 26

[35] Wetterstrand KA. **DNA Sequencing Costs: Data from the NHGRI Genome Sequencing Program (GSP)**. 2014. 6

[36] Ayman Grada and Kate Weinbrecht. **Next-Generation Sequencing: Methodology and Application**. *Journal of Investigative Dermatology*, **133**(8):e11+, August 2013. 6

[37] A VONBUBNOFF. **Next-Generation Sequencing: The Race Is On**. *Cell*, **132**(5):721–723, March 2008. 6, 9

[38] Michael Eisenstein. **The battle for sequencing supremacy**. *Nature Biotechnology*, **30**(11):1023–1026, November 2012. 8

[39] Combridge Healthtech Media Group. **Next-Generation Sequencing Survey**. 2013. 8, 11

[40] Xuming Zhou, Fengming Sun, Shixia Xu, et al. **Baiji genomes reveal low genetic variability and new insights into secondary aquatic adaptations**. *Nature Communications*, **4**, Oct 2013. 9

[41] Ningjia He, Chi Zhang, Xiwu Qi, et al. **Draft genome sequence of the mulberry tree Morus notabilis**. *Nature Communications*, **4**, September 2013. 9

[42] Cliff Meldrum, Maria A. Doyle, and Richard W. Tothill. **Next-generation sequencing for cancer diagnostics: a practical perspective.** *The Clinical biochemist. Reviews / Australian Association of Clinical Biochemists*, **32**(4):177–195, November 2011. 9

[43] Jeffrey S. Ross and Maureen Cronin. **Whole cancer genome sequencing by next-generation methods.** *American journal of clinical pathology*, **136**(4):527–539, October 2011. 9

[44] Mark I. McCarthy, Goncalo R. Abecasis, Lon R. Cardon, et al. **Genome-wide association studies for complex traits: consensus, uncertainty and challenges**. *Nature Reviews Genetics*, **9**(5):356–369, May 2008. 9

[45] John A A. Todd, Neil M M. Walker, Jason D D. Cooper, et al. **Robust associations of four new chromosome regions from genome-wide analyses of type 1 diabetes.** *Nat Genet*, June 2007. 9

[46] Robert Sladek, Ghislain Rocheleau, Johan Rung, et al. **A genome-wide association study identifies novel risk loci for type 2 diabetes**. *Nature*, **445**(7130):881–885, February 2007. 9

[47] Eleftheria Zeggini, Laura J. Scott, Richa Saxena, et al. **Meta-analysis of genome-wide association data and large-scale replication identifies additional susceptibility loci for type 2 diabetes**. *Nature Genetics*, **40**(5):638–645, March 2008. 9

[48] Christopher A. Maher, Chandan Kumar-Sinha, Xuhong Cao, et al. **Transcriptome sequencing to detect gene fusions in cancer.** *Nature*, **458**(7234):97–101, March 2009. 10

[49] Andrea L. Harper, Martin Trick, Janet Higgins, et al. **Associative transcriptomics of traits in the polyploid crop species Brassica napus**. *Nature Biotechnology*, **30**(8):798–802, July 2012. 10

[50] Andrea L. Harper, Martin Trick, Janet Higgins, et al. **Associative transcriptomics of traits in the polyploid crop species Brassica napus**. *Nature Biotechnology*, **30**(8):798–802, July 2012. 10

[51] Christoph D. Schmid and Philipp Bucher. **ChIP-Seq data reveal nucleosome architecture of human promoters.** *Cell*, **131**(5):831–832, November 2007. 10

[52] Artem Barski, Suresh Cuddapah, Kairong Cui, et al. **High-resolution profiling of histone methylations in the human genome.** *Cell*, **129**(4):823–837, May 2007. 10

[53] Stephen G. Landt, Georgi K. Marinov, Anshul Kundaje, et al. **ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia**. *Genome Research*, **22**(9):1813–1831, September 2012. 10

[54] Lin Liu, Yinhu Li, Siliang Li, Ni Hu, Yimin He, Ray Pong, Danni Lin, Lihua Lu, and Maggie Law. **Comparison of Next-Generation Sequencing Systems**. *Journal of Biomedicine and Biotechnology*, **2012**:1–11, 2012. 11

[55] Konstantinos Liolios, Nektarios Tavernarakis, Philip Hugenholtz, and Nikos C. Kyrpides. **The Genomes On Line Database (GOLD) v.2: a monitor of genome projects worldwide**. *Nucleic Acids Research*, **34**(suppl 1):D332–D334, January 2006. 12

[56] J. D. Watson and F. H. C. Crick. **Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid**. *Nature*, **171**(4356):737–738, April 1953. 14

[57] A. Brookes. **The essence of SNPs**. *Gene*, **234**(2):177–186, July 1999. 16, 83

[58] Shih-Chieh Su, C.-C. Jay Kuo, and Ting Chen. **Single nucleotide polymorphism data analysis - State-of-the-art review on this emerging field from a signal processing viewpoint**. *Signal Processing Magazine, IEEE*, **24**(1):75 –82, 01 2007. 16, 83

[59] Monica Singh, Puneetpal Singh, Pawan Juneja, et al. **SNPSNP interactions within APOE gene influence plasma lipids in postmenopausal osteoporosis**. *Rheumatology International*, March 2010. 16

[60] Adrien Coulet, Malika Smaïl-Tabbone, Pascale Benlian, et al. **SNP-Converter: An Ontology-Based Solution to Reconcile Heterogeneous SNP Descriptions for Pharmacogenomic Studies**. *Data Integration in the Life Sciences*, **4075**:82–93, 2006. 16, 83, 84, 87, 89

[61] Mark B. Gerstein, Can Bruce, Joel S. Rozowsky, et al. **What is a gene, post-ENCODE? History and updated definition**. *Genome Research*, **17**(6):669–681, June 2007. 16

[62] Helen Pearson. **Genetics: What is a gene?** *Nature*, **441**(7092):398–401, May 2006. 16

[63] Jean-Michel Claverie. **Fewer Genes, More Noncoding RNA**. *Science*, **309**(5740):1529–1530, September 2005. 17

[64] P. Carninci and Y. Hayashizaki. **Noncoding RNA transcription beyond annotated genes**. *Curr Opin Genet Dev*, **17**(2):139–44, April 2007. 17

[65] Fatih Ozsolak and Patrice M. Milos. **RNA sequencing: advances, challenges and opportunities.** *Nature reviews. Genetics*, **12**(2):87–98, February 2011. 19, 36

[66] Ryan Lister, Ronan C. O'Malley, Julian Tonti-Filippini, et al. **Highly integrated single-base resolution maps of the epigenome in Arabidopsis.** *Cell*, **133**(3):523–536, May 2008. 19

[67] Ugrappa Nagalakshmi, Zhong Wang, Karl Waern, et al. **The Transcriptional Landscape of the Yeast Genome Defined by RNA Sequencing**. *Science*, **320**(5881):1344–1349, June 2008. 19

[68] Ali Mortazavi, Brian A. Williams, Kenneth McCue, et al. **Mapping and quantifying mammalian transcriptomes by RNA-Seq.** *Nature methods*, **5**(7):621–628, July 2008. 19, 31

[69] Nicole Cloonan, Alistair R. R. Forrest, Gabriel Kolle, et al. **Stem cell transcriptome profiling via massive-scale mRNA sequencing**. *Nature Methods*, **5**(7):613–619, May 2008. 19, 21

[70] **Whole transcriptome sequencing of normal and tumor bladder tissue samples**. *Genome Biol*, **12**:23, Sep 2011. 19

[71] Erwin L. van Dijk, Yan Jaszczyszyn, and Claude Thermes. **Library preparation methods for next-generation sequencing: Tone down the bias.** *Experimental cell research*, January 2014. 21

[72] Carsten A. Raabe, Thean-Hock Tang, Juergen Brosius, and Timofey S. Rozhdestvensky. **Biases in small RNA deep sequencing data**. *Nucleic Acids Research*, **42**(3):1414–1426, February 2014. 21

[73] Young-Kook K. Kim, Jinah Yeo, Boseon Kim, Minju Ha, and V. Narry Kim. **Short Structured RNAs with Low GC Content Are Selectively Lost during Extraction from a Small Number of Cells.** *Molecular cell*, **46**(6):893–895, June 2012. 21

[74] Hai-Son Le, Marcel H. Schulz, Brenna M. McCauley, et al. **Probabilistic error correction for RNA sequencing**. *Nucleic Acids Research*, **41**(10):e109, May 2013. 22

[75] Heng Li, Jue Ruan, and Richard Durbin. **Mapping short DNA sequencing reads and calling variants using mapping quality scores.** *Genome research*, **18**(11):1851–1858, November 2008. 22

[76] Ben Langmead and Steven L. Salzberg. **Fast gapped-read alignment with Bowtie 2.** *Nature Methods*, **9**(4):357–359, April 2012. 22

[77] Heng Li and Richard Durbin. **Fast and accurate short read alignment with Burrows-Wheeler transform**. *Bioinformatics*, **25**(14):1754–1760, 2009. 22, 41, 90

[78] W. James Kent. **BLAT-The BLAST-Like Alignment Tool**. *Genome Research*, **12**(4):656–664, April 2002. 22, 33, 43, 67, 68

[79] Santiago Marco-Sola, Michael Sammeth, Roderic Guigo, and Paolo Ribeca. **The GEM mapper: fast, accurate and versatile alignment by filtration**. *Nat Meth*, **9**(12):1185–1188, December 2012. 22

[80] Kai Wang, Darshan Singh, Zheng Zeng, et al. **MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery**. *Nucleic Acids Research*, **38**(18):e178, October 2010. 22

[81] Cole Trapnell, Lior Pachter, and Steven L. Salzberg. **TopHat: discovering splice junctions with RNA-Seq**. *Bioinformatics*, **25**(9):1105–1111, May 2009. 22

[82] Melissa J. Fullwood, Chia-Lin Wei, Edison T. Liu, and Yijun Ruan. **Next-generation DNA sequencing of paired-end tags (PET) for transcriptome and genome analyses**. *Genome Research*, **19**(4):521–532, April 2009. 23

[83] Yang Li, Jeremy Chien, David I. Smith, and Jian Ma. **FusionHunter: identifying fusion transcripts in cancer using paired-end RNA-seq.** *Bioinformatics (Oxford, England)*, **27**(12):1708–1710, June 2011. 23

[84] John Eid, Adrian Fehr, Jeremy Gray, et al. **Real-Time DNA Sequencing from Single Polymerase Molecules**. *Science*, **323**(5910):133–138, January 2009. 23

[85] Elizabeth Tseng and Jason G. Underwood. **Full Length cDNA Sequencing on the PacBio RS**. *J Biomol Tech*, **24**(Suppl):S45, May 2013. 23

[86] Kin F. Au, Vittorio Sebastiano, Pegah T. Afshar, et al. **Characterization of the human ESC transcriptome by hybrid sequencing**. *Proceedings of the National Academy of Sciences*, November 2013. 23, 25

[87] Marco Ferrarini, Marco Moretto, Judson Ward, et al. **An evaluation of the PacBio RS platform for sequencing and de novo assembly of a chloroplast genome**. *BMC Genomics*, **14**(1):670+, October 2013. 23

[88] Jonathan Butler, Iain MacCallum, Michael Kleber, et al. **ALLPATHS: de novo assembly of whole-genome shotgun microreads.** *Genome research*, **18**(5):810–820, May 2008. 26, 30, 32

[89] Daniel R. Zerbino and Ewan Birney. **Velvet: Algorithms for de novo short read assembly using de Bruijn graphs**. *Genome Research*, **18**(5):821–829, May 2008. 26, 27, 30, 31

[90] Jared T. Simpson, Kim Wong, Shaun D. Jackman, et al. **ABySS: A parallel assembler for short read sequence data**. *Genome Research*, **19**(6):1117–1123, June 2009. 26, 30

[91] Pramila N. Ariyaratne and Wing-Kin Sung. **PE-Assembler: de novo assembler using short paired-end reads**. *Bioinformatics*, **27**(2):167–174, January 2011. 26, 32, 50

[92] Joshua Z. Levin, Moran Yassour, Xian Adiconis, et al. **Comprehensive comparative analysis of strand-specific RNA sequencing methods.** *Nature methods*, **7**(9):709–715, September 2010. 27

[93] Y. Fukuda, T. Washio, and M. Tomita. **Comparative study of overlapping genes in the genomes of Mycoplasma genitalium and Mycoplasma pneumoniae**. *Nucl. Acids Res.*, **27**(8):1847–1853, April 1999. 27

[94] Zackary I. Johnson and Sallie W. Chisholm. **Properties of overlapping genes are conserved across microbial genomes**. *Genome Research*, **14**(11):2268–2272, November 2004. 27

[95] Mitchell Guttman, Manuel Garber, Joshua Z. Levin, et al. **Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs**. *Nature Biotechnology*, **28**(5):503–510, May 2010. 27, 31

[96] Cole Trapnell, Lior Pachter, and Steven L. Salzberg. **TopHat: discovering splice junctions with RNA-Seq**. *Bioinformatics*, **25**(9):1105–1111, May 2009. 27

[97] Kin Fai F. Au, Hui Jiang, Lan Lin, et al. **Detection of splice junctions from paired-end RNA-seq data by SpliceMap.** *Nucleic acids research*, **38**(14):4570–4578, August 2010. 27

[98] Kai Wang, Darshan Singh, Zheng Zeng, et al. **MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery**. *Nucleic Acids Research*, **38**(18):e178, October 2010. 27

[99] Thomas D. Wu and Serban Nacu. **Fast and SNP-tolerant detection of complex variants and splicing in short reads**. *Bioinformatics*, **26**(7):873–881, April 2010. 27

[100] Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. **An Eulerian path approach to DNA fragment assembly**. *Proceedings of the National Academy of Sciences*, **98**(17):9748–9753, August 2001. 27, 30, 32

[101] Pavel A. Pevzner and Haixu Tang. **Fragment assembly with double-barreled data**. *Bioinformatics*, **17**(suppl 1):S225–S233, June 2001. 27

[102] Phillip E. C. Compeau, Pavel A. Pevzner, and Glenn Tesler. **How to apply de Bruijn graphs to genome assembly**. *Nature Biotechnology*, **29**(11):987–991, November 2011. 27, 30

[103] Yu Peng, Henry C. M. Leung, Siu-Ming Yiu, et al. **IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels**. *Bioinformatics*, **29**(13):i326–i334, July 2013. 27, 29, 31, 34

[104] Yinlong Xie, Gengxiong Wu, Jingbo Tang, et al. **SOAPdenovo-Trans: De novo transcriptome assembly with short RNA-Seq reads**. *Bioinformatics*, pages btu077+, February 2014. 27, 29, 31

[105] Brian J. Haas and Michael C. Zody. **Advancing RNA-Seq analysis**. *Nat Biotech*, **28**(5):421–423, May 2010. 27

[106] Gordon Robertson, Jacqueline Schein, Readman Chiu, et al. **De novo assembly and analysis of RNA-seq data**. *Nature Methods*, **7**(11):909–912, October 2010. 29, 31, 32, 67

[107] Manfred G. Grabherr, Brian J. Haas, Moran Yassour, et al. **Full-length transcriptome assembly from RNA-Seq data without a reference genome**. *Nat Biotech*, **29**(7):644–652, July 2011. 29, 31, 32, 67

[108] Marcel H. Schulz, Daniel R. Zerbino, Martin Vingron, et al. **Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels.** *Bioinformatics (Oxford, England)*, **28**(8):1086–1092, April 2012. 29, 31, 41, 67

[109] De Bruijn. **A combinatorial problem**. *Nederl. Akad. Wetensch. Proceedings*, **49**:758–764, 1946. 29

[110] P. A. Pevzner. **1-Tuple DNA sequencing: computer analysis.** *Journal of biomolecular structure & dynamics*, **7**(1):63–73, August 1989. 29

[111] Steven S. Skiena. *The Algorithm Design Manual*. Springer, 2nd edition, August 2008. 30

[112] R. M. Idury and M. S. Waterman. **A new algorithm for DNA sequence assembly.** *Journal of computational biology*, **2**(2):291–306, 1995. 30

[113] Mark J. P. Chaisson, Dumitru Brinza, and Pavel A. Pevzner. **De novo fragment assembly with short mate-paired reads: Does the read length matter?** *Genome Research*, **19**(2):336–346, January 2008. 30

[114] Ruiqiang Li, Hongmei Zhu, Jue Ruan, et al. **De novo assembly of human genomes with massively parallel short read sequencing**. *Genome Research*, **20**(2):265–272, December 2009. 30

[115] Jason R. Miller, Sergey Koren, and Granger Sutton. **Assembly algorithms for next-generation sequencing data**. *Genomics*, **95**(6):315–327, June 2010. 30

[116] Konrad Paszkiewicz and David J. Studholme. **De novo assembly of short sequence reads**. *Briefings in Bioinformatics*, **11**(5):457–472, September 2010. 30

[117] Yu Peng, Henry Leung, S. Yiu, et al. **IDBA - A Practical Iterative de Bruijn Graph De Novo Assembler**. In Bonnie Berger, editor, *Proceedings of the 14th Annual international conference on Research in Computational Molecular Biology*, **6044** of *RECOMB'10*, pages 426–440, Berlin, Heidelberg, 2010. Springer-Verlag. 30

[118] Jeffrey Martin, Vincent Bruno, Zhide Fang, et al. **Rnnotator: an automated de novo transcriptome assembly pipeline from stranded RNA-Seq reads**. *BMC Genomics*, **11**(1):663+, 2010. 31, 32

[119] Yu Peng, Henry C. M. Leung, S. M. Yiu, et al. **IDBA-UD: A de Novo Assembler for Single-Cell and Metagenomic Sequencing Data with Highly Uneven Depth**. *Bioinformatics*, **28**(11):1420–1428, April 2012. 31

[120] France Denoeud, Jean M. Aury, Corinne Da Silva, et al. **Annotating genomes with massive-scale RNA sequencing**. *Genome Biology*, **9**(12):R175+, 2008. 31

[121] Wei Li, Jianxing Feng, and Tao Jiang. **IsoLasso: A LASSO Regression Approach to RNA-Seq Based Transcriptome Assembly**. *Journal of Computational Biology*, **18**(11):1693–1707, November 2011. 31

[122] Yann Surget-Groba and Juan I. Montoya-Burgos. **Optimization of de novo transcriptome assembly from next-generation sequencing data**. *Genome Research*, **20**(10):1432–1440, October 2010. 31

[123] C. E. Shannon. **Prediction and entropy of printed English**. *Bell Systems Technical Journal*, **30**:50–64, 1951. 32

[124] Paul Medvedev, Son Pham, Mark Chaisson, et al. **Paired de Bruijn Graphs: A Novel Approach for Incorporating Mate Pair Information into Genome Assemblers**. *Journal of Computational Biology*, **18**(11):1625–1634, November 2011. 34

[125] Qiong Y. Zhao, Yi Wang, Yi M. Kong, et al. **Optimizing de novo transcriptome assembly from short-read RNA-Seq data: a comparative study**. *BMC Bioinformatics*, **12**(Suppl 14):S2+, 2011. 35

[126] Barbara Feldmeyer, Christopher W. Wheat, Nicolas Krezdorn, et al. **Short read Illumina data for the de novo assembly of a non-model snail species transcriptome (Radix balthica, Basommatophora, Pulmonata), and a comparison of assembler performance.** *BMC genomics*, **12**(1):317+, June 2011. 35

[127] Jia Qian Q. Wu, Lukas Habegger, Parinya Noisa, et al. **Dynamic transcriptomes during neural differentiation of human embryonic stem cells revealed by short, long, and paired-end sequencing.** *Proceedings of the National Academy of Sciences of the United States of America*, **107**(11):5254–5259, March 2010. 36

[128] Said Assou, Imene Boumela, Delphine Haouzi, et al. **Dynamic changes in gene expression during human early embryo development: from fundamental aspects to clinical applications**. 36

[129] Daniel R. Rhodes and Arul M. Chinnaiyan. **Integrative analysis of the cancer transcriptome**. *Nature Genetics*, **37**:S31–S37, June 2005. 36

[130] Jinfeng Liu, William Lee, Zhaoshi Jiang, et al. **Genome and transcriptome sequencing of lung cancers reveal diverse mutational and splicing events**. *Genome Research*, **22**(12):2315–2327, December 2012. 36

[131] Vincent Lacroix, Michael Sammeth, Roderic Guigó, and Anne Bergeron. **Exact Transcriptome Reconstruction from Short Sequence Reads**. In *WABI*, pages 50–63, 2008. 36

[132] Yu Peng, Henry C. M. Leung, Siu-Ming Yiu, et al. **IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels**. *Bioinformatics*, **29**(13):i326–i334, July 2013. 38

[133] Nicholas Rhind, Zehua Chen, Moran Yassour, et al. **Comparative Functional Genomics of the Fission Yeasts**. *Science*, **332**(6032):930–936, May 2011. 41

[134] Z. Ning, A. J. Cox, and J. C. Mullikin. **SSAHA: a fast search method for large DNA databases.** *Genome research*, **11**(10):1725–1729, October 2001. 46

[135] Robert Tarjan. **Depth-First Search and Linear Graph Algorithms**. *SIAM Journal on Computing*, **1**(2):146–160, 1972. 63

[136] Yin Hu, Yan Huang, Ying Du, et al. **DiffSplice: the genome-wide detection of differential splicing events with RNA-seq**. *Nucleic Acids Research*, **41**(2):e39, January 2013. 63, 64, 65, 66

[137] Yan Huang, Yin Hu, Corbin D. Jones, et al. **A Robust Method for Transcript Quantification with RNA-Seq Data**. *Journal of Computational Biology*, **20**(3):167–187, March 2013. 64

[138] Oliver Stegle, Philipp Drewe, Regina Bohnert, et al. **Statistical Tests for Detecting Differential RNA-Transcript Expression from Read Counts**. *Nature Precedings*, (713), May 2010. 64

[139] Darshan Singh, Christian F. Orellana, Yin Hu, et al. **FDM: a graph-based statistical method to detect differential transcription using RNA-seq data**. *Bioinformatics*, **27**(19):2633–2640, October 2011. 64

[140] A. P. Dempster, N. M. Laird, and D. B. Rubin. **Maximum Likelihood from Incomplete Data via the EM Algorithm**. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**(1):1–38, 1977. 65

[141] Hui Jiang and Wing H. Wong. **Statistical inferences for isoform expression in RNA-Seq**. *Bioinformatics*, **25**(8):1026–1032, April 2009. 65

[142] Yohei Sasagawa, Itoshi Nikaido, Tetsutaro Hayashi, Hiroki Danno, Kenichiro Uno, Takeshi Imai, and Hiroki Ueda. **Quartz-Seq: a highly reproducible and sensitive single-cell RNA sequencing method, reveals non-genetic gene-expression heterogeneity**. *Genome Biology*, **14**(4):R31+, April 2013. 74

[143] Raymond D. Miller and Pui-Yan Kwok. **The birth and death of human single-nucleotide polymorphisms: new experimental evidence and implications for human history and medicine**. *Human Molecular Genetics*, **10**(20):2195–2198, 2001. 83

[144] K. Tamura, M. Suzuki, H. Arakawa, et al. **Linkage and Association Studies of STAT6 Gene Polymorphisms and Allergic Diseases**. *International Archives of Allergy and Immunology*, **131**(1):33–38, 2003. 83

[145] O. Horaitis and R. G. Cotton. **The challenge of documenting mutation across the genome: the human genome variation society approach**. *Hum Mutat*, **23**(5):447–452, 2004. 83, 85

[146] Elizabeth M. Smigielski, Karl Sirotkin, Minghong Ward, et al. **dbSNP: a database of single nucleotide polymorphisms**. *Nucl. Acids Res.*, **28**(1):352–355, 2000. 83

[147] A. J. Brookes, H. Lehvaslaiho, M. Siegfried, et al. **HGBASE: a database of SNPs and other variations in and around human genes**. *Nucleic Acids Res*, **28**:356–60+, 2000. 83

[148] International HapMap Consortium. **The International HapMap Project**. *Nature*, **426**(6968):789–796, December 2003. 83

[149] M Hirakawa, T Tanaka, Y Hashimoto, et al. **JSNP: a database of common gene variations in the Japanese population**. *Nucleic Acids Res*, **30**(1):158–62, 2002. 83, 85

[150] den Dunnen JT and Paalman MH. **Standardizing mutation nomenclature: why bother?** *Hum Mutat*, **22**(3):181–2, 2003. 84, 85

[151] Sharon Marsh, Pui Kwok, and Howard L. McLeod. **SNP databases and pharmacogenetics: great start, but a long way to go**. *Human Mutation*, **20**(3):174–179, 2002. 84, 85

[152] Richard G. H. Cotton. **Recommendations of the 2006 Human Variome Project meeting**. *Nat Genet*, **39**(4):433–436, Apr 2007. 84, 88

[153] Martin Wildeman, Ernest van Ophuizen, Johan T. den Dunnen, et al. **Improving sequence variant descriptions in mutation databases and literature using the Mutalyzer sequence variation nomenclature checker**. *Human Mutation*, **29**(1):6–13, 2008. 84, 85, 87, 93

[154] H. Wain, J. White, and S. Povey. **The changing challenges of nomenclature**. *Cytogenet Cell Genet*, **86**(2):162–4, 1999. 84

[155] Antonarakis SE and the Nomenclature Working Group. **Recommendations for a nomenclature system for human gene mutations**. *Hum Mutat*, **11**(1):1–3, 1998. 84

[156] J. T. den Dunnen and S. E. Antonarakis. **Mutation nomenclature extensions and suggestions to describe complex mutations: a discussion**. *Human mutation*, **15**(1):7–12, 2000. 84

[157] J. T. den Dunnen and S. E. Antonarakis. **Nomenclature for the description of human sequence variations**. *Human genetics*, **109**(1):121–124, 2001. 84

[158] Richard G. H. Cotton and Ourania Horaitis. **Quality control in the discovery, reporting, and recording of genomic variation**. *Human Mutation*, **15**(1):16–21, 2000. 85

[159] James T.L. Mah, Danny C.C. Poo, and Shaojiang Cai. **UASMAs (Universal Automated SNP Mapping Algorithms): a set of algorithms to instantaneously map SNPs in real time to aid functional SNP discovery**. *Proc. VLDB2010 Endow.*, **3**(1), 2010. 85, 90

[160] Raymond Dalgleish, Paul Flicek, Fiona Cunningham, et al. **Locus Reference Genomic sequences: an improved basis for describing human DNA variants**. *Genome Medicine*, **2**(4):24+, April 2010. 86

[161] Ivo F. A. C. Fokkema, Peter E. M. Taschner, Gerard C. P. Schaafsma, et al. **LOVD v.2.0: the next generation in gene variant databases**. *Human Mutation*, **32**(5):557–563, 2011. 86

[162] D. Fredman, G. Munns, D. Rios, et al. **HGVbase: a curated resource describing human DNA variation and phenotype relationships**. *Nucleic Acids Research*, **32**(suppl 1):D516–D519, 2004. 86

[163] Gudmundur A. Thorisson, Owen Lancaster, Robert C. Free, et al. **HGVbaseG2P: a central genetic association database**. *Nucleic Acids Research*, **37**(suppl 1):D797–D802, 2009. 86

[164] Micheal Hewett, Diane E. Oliver, Daniel L. Rubin, et al. **PharmGKB: the Pharmacogenetics Knowledge Base**. *Nucl. Acids Res.*, **30**(1):163–165, January 2002. 87

[165] A. Riva and I. S. Kohane. **SNPper: retrieval and analysis of human SNPs**. *Bioinformatics*, **18**(12):1681–1685, 2002. 87

[166] Bradley M. Hemminger, Billy Saelim, and Patrick F. Sullivan. **TAMAL: an integrated approach to choosing SNPs for genetic studies of human complex traits**. *Bioinformatics*, **22**(5):626–627, 2006. 87

[167] Rachel Karchin, Mark Diekhans, Libusha Kelly, et al. **LS-SNP: large-scale annotation of coding non-synonymous SNPs based on multiple information sources**. *Bioinformatics*, **21**(12):2814–2820, 2005. 87

[168] Jan Aerts, Yves Wetzels, Nadine Cohen, et al. **Data mining of public SNP databases for the selection of intragenic SNPs**. *Human Mutation*, **20**(3):162–173, 2002. 87

[169] Adrien Coulet, Malika Smaïl Tabbone, Pascale Benlian, et al. **SNP-Ontology for semantic integration of genomic variation data**. *14th Annual International Conference on Intelligent Systems for Molecular Biology - ISMB'06*, 08 2006. 87

[170] Walter F. Bodmer. **HLA: what's in a name?** *Tissue Antigens*, **49**(3):293–296, 1997. 88

[171] Ben Langmead, Cole Trapnell, Mihai Pop, et al. **Ultrafast and memory-efficient alignment of short DNA sequences to the human genome**. *Genome biology*, **10**(3):R25+, 2009. 90

[172] M. Burrows and D. J. Wheeler. **A block-sorting lossless data compression algorithm.** Technical Report 124, 1994. 90

[173] P. Ferragina and G. Manzini. **Opportunistic data structures with applications**. *Foundations of Computer Science, Annual IEEE Symposium on*, **0**:390–398, 2000. 90

[174] Daniel C. Richter, Felix Ott, Alexander F. Auch, et al. **MetaSimA Sequencing Simulator for Genomics and Metagenomics**. *PLoS ONE*, **3**(10):e3373, 10 2008. 93