Local assembly and pre-mRNA splicing analyses by high-throughput sequencing data

Hsien-chao Chou
Iowa State University
Local assembly and pre-mRNA splicing analyses by high-throughput sequencing data

by

Hsien-chao Chou

A dissertation submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Bioinformatics and Computational Biology

Program of Study Committee:
Volker Brendel, Co-Major Professor
Vasant Honavar, Co-Major Professor
Amy Toth
Xiaojie Huang
Bing Yang.

Iowa State University
Ames, Iowa
2012
TABLE OF CONTENTS

LIST OF FIGURES iv
LIST OF TABLES v
ABSTRACT vi

Chapter 1 General Introduction 1
   Next generation sequencing 2
   Mechanisms of pre-mRNA splicing 8
   Splice site prediction and RNA-Seq alignment 14
   Research goals 16

Chapter 2 SRAssembler: Local Assembly of Homologous Regions by Genomic DNA 18
   Reads and Homologous Genes 18
   Abstract 18
   Rationale 19
   Results and discussion 22
   Conclusions 34
   Materials and methods 35
   Availability and requirements 41

Chapter 3 Genome-Wide Survey of Stepwise Intron Removal by RNA-Seq Data 42
   Abstract 42
   Background 44
   Methods 48
   Results and discussion 52
   Conclusions 66
   Availability and requirements 66

Chapter 4 Splicing Intermediates Analyses using RNA-Seq Data 67
   Abstract 67
   Background 68
   Methods 71
   Results and discussion 75
   Conclusions 85
   Availability of supporting data 86

Chapter 5 General Conclusion 87
   Summary 87
   Recommendation for future research 88

References 91

APPENDIX A. ADDITIONAL MATERIAL 100

APPENDIX B. SRASSEMBLER INSTALLATION AND USAGE 101
   Installation 101
LIST OF FIGURES

Figure 1.1 The de Bruijn graph. 7
Figure 1.2 Spliceosomal splicing and self-splicing. 12
Figure 1.3 Alternative splicing patterns. 13
Figure 2.1 The spliced alignment of OS01G18860.2 and OS1G04560.1. 24
Figure 2.2 Chromosome walking of gene AT1G01950.1. 25
Figure 2.3 Comparison of assemblies. 31
Figure 2.4 Running time of SRAssembler. 33
Figure 2.5 In silico chromosome walking strategy. 36
Figure 2.6 Proposed pipeline of SRAssembler. 39
Figure 3.1 Long intron splicing models. 47
Figure 3.2 Confirmation of RSSs. 51
Figure 3.3 Comparison of simulated and real intron sequences. 53
Figure 3.4 The sequence logo around the RSSs in four species. 57
Figure 3.5 Information content plots. 58
Figure 3.6 Length distribution. 60
Figure 3.7 Sequence logo for different subintron length. 64
Figure 3.8 Type I and type III RSSs in the same intron. 65
Figure 4.1 The calculation of the abundances of the primary, intermediate and mature transcripts. 71
Figure 4.2 The log likelihood of AT1G11020. 78
Figure 4.3 Fragment length distribution of dataset SRR391052. 80
Figure 4.4 Results of transcript proportion estimation of dataset SRR391052. 82
Figure 4.5 Distinguishing isoforms from intermediates. 84
Figure 4.6 Fragment length distribution of dataset SRR360152. 85
Figure 5.1 The integrated mRNA processing model. 90
LIST OF TABLES

Table 2.1 - The simulation results.  
Table 2.2 - The results of 11 Arabidopsis genes.  
Table 2.3 - The results of assembling gene family of peroxisomal biogenesis factor.  
Table 3.1 - Datasets  
Table 3.2 - The candidate sites for type I, type II and type III RSSs. 
Table 3.3 - Confirmed sites.  
Table 3.4 - Descriptive statistics for length distribution.  
Table 3.5 - The experimental validated sites found by RSSFinder.  
Table 4.1 - The length and proportion specified in the simulation of gene AT1G11020.  
Table 4.2 - The simulation results using uniquely mapped reads with different coverage.  
Table 4.3 - The estimation of $\theta$ for gene AT1G11020.  
Table 4.4 - The simulation results using E-M algorithm with different coverage.  
Table 4.5 - The estimation of $\theta$ for gene AT1G11020.
ABSTRACT

Next generation sequencing (NGS) approaches have become one of the most widely used tools in biotechnology. With high throughput sequencing, people can analyze non-model species at an unprecedented high resolution. NGS provides fast, deep and cheap sequencing solutions, and it has been used to answer various biological questions. In this thesis, I have developed a set of tools and used them to study several interesting research topics. First, *de novo* whole-genome assembly is still a very challenging technical task. For eukaryotic genomes, *de novo* assembly typically requires computational resources with very large memory and fast processors. Instead of trying to assemble the whole genome as done in previous approaches, I focus on efficiently reconstructing the genomic regions related to the homologous protein or cDNA sequences. I have developed SRAssembler, a local assembly program using the iterative chromosome walking strategy to assemble the loci of interest directly. Second, I used high-throughput RNA sequencing (referred to as RNA-Seq) data to analyze different intron splicing models and their relative frequency of occurrence. The first mechanism I explored is the recursive splicing patterns in large introns. I have implemented a pipeline called RSSFinder, which can search for recursive sites confirmed by RNA-Seq data. My study suggests the prevalence of recursive splicing in different species. These predicted recursive sites can also be used to investigate certain diseases associated with abnormal splicing of transcripts. In addition, I have demonstrated the use of RNA-Seq data to decipher the detailed mechanisms involved in splicing and their relationship with transcription. Here I proposed mathematical models to estimate the distribution of mRNA splicing intermediates. I
evaluated my models with simulated data and an Arabidopsis thaliana dataset. My results indicate that co-transcriptional splicing is widespread in Arabidopsis thaliana.
CHAPTER 1 GENERAL INTRODUCTION

In 1990, an ambitious project called Human Genome Project [1] had started a new era of genome biology. The original plan for this project was conservative and laborious. In order to get the correct sequence order, each piece of DNA sequence must be mapped to markers or signposts before the targets could be sequenced. In 1998, a new approach named shotgun sequencing was proposed by Craig Venter [2], and he announced that the human genome draft would be finished by the end of 2000. This method uses computer programs to piece the fragments together by finding the overlapping sequences. Venter finished the first draft on June 26, 2000, demonstrating the power of automatic computer assembly programs. Based on the idea of shotgun sequencing, a new parallel sequencing technology has been introduced for genome analysis. This so-called next generation sequencing (NGS) technologies has revolutionized the genomic research. NGS provides fast, deep and cheap sequencing solutions, and it has been widely used to answer various kinds of biological questions. NGS also promotes the studies of non-model species at very low costs. The genome of these species can be assembled using NGS de novo assembly programs. NGS is also used to sequence RNA molecules, also known as RNA-Seq. In addition to the gene expression studies, RNA-Seq can be used to examine the microcosmic mechanisms in transcriptome such as small nuclear RNAs or intron splicing thanks to the depth of RNA-Seq reads.

In this thesis, I answer some computational and biological questions using NGS technologies. The first question is that instead of assembly whole genome, how can we efficiently assemble the interested genes only? The fundamental problem of whole genome
de novo assembly is that it requires very expensive computer resources with very large memory (hundreds of GB). In many cases, biologists are interested in a few genes only. For this purpose, I propose a local assembly algorithm which can efficiently assemble the loci of interest with lower computational resources. The second question is how are the long introns spliced? I use the RNA-Seq reads to search for the evidence of recursive splicing, which is a stepwise intron removal mechanism proposed by [3]. The third question is whether splicing is co-transcriptional? How are the relative abundances distributed among the splicing intermediates? Since RNA-Seq is so deep that many reads are mapped to introns or exon/intron junctions, I can take advantage of this information to estimate the distribution of mRNA intermediates using mathematical models.

The rest part of this chapter will focus on the background information about NGS and splicing mechanisms. I will also describe the splice site prediction tools as well. Chapter 2 is focused on the local assembly algorithm using genomic reads. Chapter 3 and 4 are focused on intron splicing using RNA-Seq datasets.

Next generation sequencing

Next generation sequencing (NGS) has become one of the most promising bioinformatics technologies over the past five years. In this section, I will describe the background information about NGS platforms, basic strategies dealing with NGS reads (mapping and assembly) and its applications.
NGS platforms

The first NGS system was introduced by Roche 454 [4] in 2005. Illumina GA [5] and ABI SOLiD (Sequencing by Oligo Ligation Detection) [6] are then released in the following years. These three platforms are the most typical and popular NGS platforms. The advantage of 454 system is the sequencing speed and the longer read length. The read length of first generation 454 machine is around 150bp. The newer 454 GS FLX system can generate one million reads with read length 700bp in one day. Another well-known NGS platform is SOLiD, which uses unique two-base color space coding. Each base is checked twice therefore producing more accurate base-calling. SOLiDv4 can sequence up to 1.4 billion paired-end reads (the fragments are sequenced from both ends) in two weeks. Illumina offers highest throughput. For example, Illumina HiSeq2000 can produce 600Gb per run with read length 100bp in 8 days. The cost of HiSeq 2000 is only $0.02 per million bases [7], which is cheapest compared with 454 and SOLiD. A newer MiSeq sequencer was released in 2011. It could produce 150bp paired-end reads with 1.5Gb per run in 10 hours including the library preparation time. MiSeq and Ion Personal Genome Machine (PGM) are targeted to clinical applications and small labs.

One of the major problems of NGS is the sequencing error. Like Sanger sequencing, NGS sequencers also provide the quality score indicating the probabilities of base error using phred algorithm [8]. This algorithm assigns a quality value for each base. In Sanger phred, the quality scores is calculated by \( Q = -10\log_{10}p \), where \( p \) is the probability of the base call is incorrect. Each sequencing platform has different error profiles. In Sanger phred, the quality scores are from 0 to 93, and they can be represented by ASCII code 33-126 in plain text sequence files. The quality scores range in Illumina 1.0 is from -5 to 40 (ASCII code 59-
104). But the quality values range has returned to Sanger format after Illumina 1.8. The SOLiD system uses Sanger quality scores based on the color call of two bases. The range is from 0 to 45.

Since Illumina sequencers can generate highest throughput of NGS reads, they have become the most dominant platform in this field. One of the main problems of Illumina reads is the read length. In the library preparation step, the DNA or RNA molecules are chopped into smaller fragments. Each fragment can be sequenced from one end up to 150bp only. The first form of Illumina reads is single-end. That is, only one end of the fragment can be sequenced. The major problem of single end reads is the ambiguity when reads are mapped to multiple loci. A simple improvement to the single-end library preparation is to sequence both ends of fragments (scanning both the forward and reverse template strand). The paired-end sequencing incorporates the fragment length information which can significantly improve the mapping and assembly accuracy. The typical fragment length of paired-end sequencing is 200-500bp. In terms of genomic assembly, this fragment length is still too short when scaffolding contigs. A newer library preparation method can produce paired reads (referred to as mate pairs) separated by longer distance (2kb ~ 8kb). In this method, longer fragments are circularized and both ends are sequenced. Mate pairs reads are very useful for the de novo genome.

**Mapping strategies**

If the reference genome is available, the straightforward way to deal with NGS reads is to map the reads back to the reference sequences. Sequence alignment is an old bioinformatics problem. The classical method is to align reads back to genome using
dynamic programming such as Smith-Waterman algorithm [9]. The drawback of dynamic programming is the speed. Some faster alignment algorithms such as BLAST indexes k-mers and they can efficiently pinpoint the location. These k-mer seed sequences are then extended using traditional alignment methods [10]. But these methods are not designed to handle millions of very short reads. Therefore, the new NGS aligners are rapidly introduced and have become one of the prosperous fields in bioinformatics. To promote the aligning efficiency, these aligners only allow small number of mismatches, insertions and deletions. For example, for the read length 36bp, a read can be split into 4 sub-reads with 9bp. If only two mismatches are allowed, two out of four sub-reads must be exactly matched. Therefore, the potential matched reads can be efficiently identified. ELAND [11], ZOOM [12], SeqMap [13] and SOAP [14] are all the aligners based on such principle by building a hash table of substring of length k from either for the reads or the reference sequences.

The main drawback of these programs is the requirement of memory. Bowtie [15], one of the most successful NGS aligner, uses Burrows-Wheeler transform (BWT) based on full-text minute-space (FM) index. The size of FM index is small enough to load it to the memory of a typical personal computer. For example, the memory footprint for human genome needs only about 1.3 GB RAM. The newer version of SOAP [16] and BWA [17] are also used BWT as the indexing strategy.

**Assembly strategies**

If the reference genome is not available, the NGS reads must be assembled directly. The biggest challenge of *de novo* assembly is the short read length. If the target regions contain repeat sequences, then these regions become indistinguishable. In fact, the assembly
programs of whole genome shotgun (WGS) sequencing were developed before the advent of NGS. But these conventional assemblers such as PCAP [18], Arachne [19] or Celera Assembler [20] are unable to deal with the massive number of microreads. These assemblers employ the Overlap/Layout/Consensus (OLC) three phases approach to find the consensus sequences. The overlap phase involves the pair-wise comparison of all reads using pre-computed $k$-mer seeds. Then the approximate read layout is constructed by the overlap graph. Finally, the multiple sequence alignment is used to determine the precise read layout, thereby obtaining the final consensus sequences. This method can still be used to assemble longer and low throughput NGS reads. For example Newbler [21] is a very successful assembler based on this approach, which is widely used to assemble 454 reads.

To assemble large number reads produced by Illumina or SOLiD sequencers, many assemblers have been developed based on the de Bruijn graphs [22] such as Velvet [23], ABysS [24], ALLPATH [25], and SOAPdenovo [26]. The principle of de Bruijn graph is to represent a sequence as a set of $k$-mer components. Figure 1.1 demonstrates examples of the de Bruijn graph. In these examples, each node represents a 10-mer sequence. If the 10-mer sequence can be found in two sequences, the node will be shared by them. When $k$ is very large, the more memory is required to process and store the graph.

For de novo genome assembly, fragment length information is crucial to obtain quality results. The better strategy is to assemble multiple libraries including mate pair reads. Such tasks require extremely high computer resources. For example ALLPATH and Velvet need hundreds of GB RAM to assemble eukaryotic genomes using multiple libraries.
Assemblies are usually evaluated by the size of assembled contigs including average contig length, maximum contig length, total contig length and N50 length. The N50 is the shortest contig length in the set representing at least 50% combined length of the assembly.

Choosing the best parameters is the key to the success of the assemblies. For example, different k-mer may produce totally different assemblies. One may need to try different k-mer and select the assembly producing the best N50 value.

(A) NGS Read: ATGATACGGCGACCACG

(B) NGS Read1: ATGATACGGCGAC
NGS Read2: GCGATACGGCGAG

Figure 1.1 The de Bruijn graph.
(A) A NGS read can be represented as a set of k-mer substrings. In this example, k=10. Each 10-mer sequence is denoted as a node in the de Bruijn graph. (B) If a 10-mer sequence can be found in two sequences, the node will be shared by them.

In addition to the genome assembly, RNA-Seq reads can be assembled without a reference genome. The transcriptome de novo assembly is even more challenging than the genome assembly because the coverage of RNA-Seq is not evenly distributed. When the coverage is too low, higher k-mer may yield poor results. Therefore, a single k-mer may not
generate an optimal assembly. To overcome this problem, several transcriptome assembly programs such Trans-ABySS [27] and Velvet/Oases [28] can produce a set of assemblies using different $k$, and then merge them together to get the best assembly. Like whole-genome assembling, running the transcriptome assembling also requires very large memory. For example, Trinity [29] needs around 1GB RAM to process one million reads.

**NGS applications**

Applications of the NGS technologies include the genetic variation detection [30-32], DNA methylation [33-35] and Chip-Seq [36-38]. Other applications have been used to sequence mRNA and small nuclear RNAs and allow global measurement of transcript abundances. Before the development of RNA-Seq, transcriptome analysis [39] relied on the hybridization based[40, 41] or tag sequence-based approaches [42]. RNA-Seq has several advantages over these approaches. For example, RNA-Seq can be used to detect novel genes or transcripts [43-47]. RNA-Seq also has lower background signal and no upper limit for quantification, therefore getting a larger dynamic range of expression levels than microarray analyses [48].

**Mechanisms of pre-mRNA splicing**

**Split genes**

Introns were first found by Philip Sharp in 1977 [49]. In their experiments with adenoviruses, they hybridized RNA to the DNA template and inspected them by electron microscopy. Their found that the RNA-DNA hybrids are interrupted by DNA loops, suggesting the information encoded in genes is not continuous. The coding and non-coding
part of genes is referred to as exons and introns respectively. When mRNA is synthesized in eukaryotes, the mRNA precursors still contain introns transcribed from DNA template. The introns are then removed and adjacent exons are ligated. The origin of introns is still not completely known. At first, processing these useless components appears very wasteful in terms of energy consumption, but today we know the existence of introns largely facilitates the diversity of gene products. The intron removal may involve different pathways, thereby producing functional distinct mRNA isoforms from a single gene. This mechanism is known as alternative splicing. Exons often encode independent functional domains. Therefore, alternative splicing provides a complex design to assemble different functional modules. This is an economic way to achieve the proteome diversity. Furthermore, introns can also play the cis-regulatory roles in splicing. For example, the intronic enhancers and silencers can promote or inhibit the splice site recognition. The length of introns affects the efficiency of transcription as well, and then the gene expression can be regulated.

**Pre-mRNA splicing**

To accurately splice the primary mRNA molecules, the exon and intron boundary must be correctly recognized first. The observation of intron sequences reveals that almost all introns begin with dinucleotide GU (5’ donor site) and end with dinucleotide AG (3’ acceptor site). This GU/AG motif is also known as core splicing signals. The branch point, an adenine nucleotide, is a key position to form the splicing lariat (described later), is located at 20-100bp upstream of the acceptor sites. The mammalian consensus intron sequence can be denoted as:

\[
5'-AG/GUAAGU-intron-YNCURAC-Y_nNAG/G-3'
\]
where exon/intron boundary is denoted by slashes. Y is pyrimidine, R is purine, A is the branch point, and N is any base. Note \( Y_n \) is a stretch of pyrimidine known as polypyrimidine tract, which is known to promote the recognition of acceptor sites.

Most intron removal is catalyzed by spliceosome, a complex containing small nuclear ribonucleoproteins (snRNPs) and other protein factors. snRNPs contain the RNA molecules which can pair with mRNA sequences. There are two types of spliceosomal splicing pathways: the canonical and non-canonical. The canonical pathway accounts for 99% of splicing. The aforementioned GU/AG is the motif sequence of canonical splicing. The canonical spliceosomal splicing cycle begins with the recognition of 5’ splice site by U1 snRNP. The branch point is then bound by the splicing factor SF1. The U2 Auxiliary Factor (U2AF) also interacts with 3’ splice site and the polypyrimidine tract. Next, the U2 snRNP binds to the branch-point and join to form the “A complex”. Another U4/U5/U6 tri-snRNP then joins and forms “B1 complex”. Later, U1 is displaced by U6, which pairs with U2. At this point, U1 and U4 are dissociated and the “B2 complex” forms. The 2’OH of the branch-point attacks on the 5’ splice site via transesterification reaction and forms the lariat splicing intermediate and the “C1 complex”. The second transesterification reaction is performed by the attack of free 3’OH of upstream exon on the last nucleotide of the intron, forming the “C2 complex”. The C2 complex joins the exons and releases the lariat, which is then degraded.

In the non-canonical pathway, the motif of exon/intron boundary is AU/AC. Since the consensus intron sequence is different, the splice site recognition is done by another group of spliceosomes. The roles of U1, U2, U4 and U6 is replaced by U11, U12, U4atac and U6atac snRNPs respectively.
In addition to the spliceosomal introns, some introns can be spliced without the help of a spliceosome. These self-splicing RNAs can be classified into two groups. The group I introns was first discovered in 26S rRNA gene of Tetrahymena by Thomas Cech [50]. Group I self-splicing occurs via the process that a guanine nucleotide in the intron attacks adenine nucleotide at the 5’ end. Then the OH group at the upstream exon attacks the downstream exon, releasing the linear intron. The group II self-splicing acts very similar to the spliceosomal splicing, including the transesterification by the branch-point. The introns then form a secondary structure akin to the spliceosomal lariats (Figure 1.2).

Since the motif of donor and acceptor splice sites has very low information content. The true splice sites must compete with many false ones. This problem may become even worse when the intron size is very large. In fact, the splice site recognition is promoted by a group of extrinsic factors such as SR proteins or hnRNP (heterogeneous nuclear ribonucleoparticles) proteins. SR proteins contain one or more serine and arginine-rich domains and can interact with RNA via RNA Recognition Motifs (RRMs). The cis-acting splicing regulatory elements such as exonic splicing enhancers (ESEs) or exonic splicing silencers (ESSs) can interact with SR proteins and hnRNP proteins to promote and inhibit the splicing.

As mentioned earlier, the splice sites can be recognized by the ends of introns. This is also known as “intron definition”. Today we know the mechanism of intron definition is prevalent in shorter introns. The splice sites can be bridged across the introns by the spliceosomal components and the extrinsic proteins. In longer introns, however, some splice sites can be recognized by both ends of exons (referred as to “exon definition”)[51]. In this case, mutation of the splice sites will cause intron retention instead of exon skipping.
Both spliceosomal and group II self-splicing involve the attack of OH group at branch-point. The group I splicing shows a distinct splicing mechanism contains the attack of a guanine nucleotide[52].

**Alternative splicing**

Some genes can produce more than one isoforms because of different splicing patterns. This mechanism is also known as alternative splicing. Alternative splicing is prevalent in eukaryotic genes. For example, a human transcriptome study by high throughput sequencing indicates that more than 95% human genes undergo alternative splicing [53].

Alternative splicing can be classified into five categories (see **Figure 1.3**):

1. Alternative donor site: An alternative upstream exon boundary used.
2. Alternative acceptor site: An alternative downstream exon boundary used.
3. Exon skipping: An exon may be retained or removed.

4. Intron retention: An intron may be retained or removed.

5. Mutually exclusive exons: Only one of two exons is retained.

Figure 1.3 Alternative splicing patterns.
(A) Alternative donor site. (B) Alternative acceptor site. (C) Exon skipping. (D) Intron retention. (E) Mutually exclusive exons.

Different alternative splicing patterns can be regulated by trans-acting proteins or cis-acting elements. For example, some intronic sequences may encode microRNAs, which may inhibit the isoform expression via RNA interference (RNAi) pathways [54]. In some cases,
alternative splicing may disrupt the open read frame by introducing frameshifts or early immature stop-codon. The mRNAs with abnormal stop-codons may be degraded through the nonsense mediated decay (NMD) pathway [55], but other abnormal splicing might contribute to diseases or genetic disorder [56, 57]. In short, alternative splicing plays a crucial role in biodiversity, diseases, gene expression regulation.

**Splice site prediction and RNA-Seq alignment**

**Splice site prediction**

After part of splicing mechanism has been deciphered, several tools have been developed to find the splice junctions given a piece of DNA sequence. These tools typically use the context information around the known splice sites to predict new splice sites from the genomic sequences alone, without mRNA evidence. For example, SplicePredictor [58] predicts splice sites by measuring the intrinsic splice site quality [59], local optimality of the site, and the contribution of the sites to the splicing patterns in the context of the flanking sequence segments. GeneSplicer [60] uses similar Markov modeling to predict Arabidopsis thaliana and human splice sites. Several machine learning based splice site detectors are also reported to perform at a very good level [61-65].

**Spliced alignment for RNA-Seq data**

Traditionally, transcript discovery and gene annotation have relied on the alignment of full-length cDNAs or ESTs as well as homologous protein sequences to the genomic sequences being annotated. Such spliced alignment tools include NAP [66], Genewise [67], Blat [68], GeneSeqer [69] and GMAP [70]. This strategy allows long alignment gaps in the
mapped transcripts or proteins that correspond to introns in the reference genome. However, traditional spliced alignment methods are not directly applicable to RNA-Seq because they need longer sequences that cover the exon-exon boundary for the reliable gene structure prediction. For example, suppose a read maps on the exon-exon junction in such a way that only few bases are within one of the exons. In this case, traditional dynamic programming methods trying to make the spliced alignment across a potential long intron are ineffective. Aforementioned NGS aligners can rapidly map reads to reference genomes. But these tools allow only small number of mismatches, insertions and deletions. Some NGS spliced alignment tools such as Tophat [43, 45] or G.Mo.R-Se [43] assemble all mapped reads into potential exon islands. The candidate exon junctions could then be obtained by searching within a specific window around the island boundary, say 100 nucleotides. The possible donor and acceptor sites could be determined by the terminal dinucleotide pairs, such as canonical GT-AG splice sites. All possible splice sites are tested by the reads that were not mapped initially. However, these methods may incur too many false-positives. On the other hand, algorithms like ERANGE [71] or RNA-mate [72] use a known annotated junction libraries. Therefore, this method is generally limited to the cases that we only intend to know the expression levels instead of discovering novel junctions. Other spliced alignment tools use the splice site prediction models to improve the accuracy of the junction discovery. For example, QPALMA [44] applies machine learning techniques to train a support vector machine from known splice junctions. GSNAP [73] can evaluate the surrounding genomic sequence using probabilistic models of splice sites.
Research goals

The work in this thesis focuses on using NGS data to answer several computational and biological questions. Below I introduce the goals of my three main topics. In chapter 2, 3 and 4, I will describe my methods and results for them in detail.

Local assembly of homologous regions by genomic DNA reads

For eukaryotic genomes, de novo assembly typically requires the computer resources with very large memory and fast processors. Even with such extensive computational resources, an assembly may still take several days to finish. However, sometimes biologists are only interested in small set of genes with known homologous protein sequences. My goal is to design an algorithm to quickly and accurately assemble the loci of interests. I have developed a program called SRAssembler (Selective and Recursive local Assembler), which is shown to be able to do the local assembling efficiently using iterative chromosome walking strategy.

Genome-wide survey of stepwise intron removal by RNA-Seq data

Stepwise intron removal has been shown an important mechanism in Drosophila. However, only very small number of recursive sites can be validated. My goal is to use RNA-Seq data to search for the stepwise intron removal evidence including two types of recursive splicing and intrasplicing. I have developed a pipeline called RSSFinder (Recursive Splice Sites Finder) to investigate the recursive splicing mechanisms in four species including Drosophila, mouse, rice and Arabidopsis.
Splicing intermediates analysis using RNA-Seq data

The mechanisms of transcription and splicing are two interactive processes. The splicing pathways could be very complex with many possible mRNA intermediates. Since RNA-Seq data is so deep that many reads are mapped on intronic regions, I can use the RNA-Seq reads to estimate the distribution of mRNA intermediates. I used simulation data to evaluate the performance of my algorithms. I also use this model to examine the Arabidopsis genes with 3 and 4 exons. My study may serve as the foundation to further understand the splicing dynamics and the relationship between splicing and transcription.
CHAPTER 2 SRASSEMBLER: LOCAL ASSEMBLY OF
HOMOLOGOUS REGIONS BY GENOMIC DNA READS AND
HOMOLOGOUS GENES

A paper to be submitted to Genome Biology

Hsien-chao Chou¹, Volker P. Brendel ²,³*

Abstract

We have developed a tool, SRAssembler, which locally and recursively assembles the
genomic reads associated with the homologous query genes. We demonstrate SRAssembler
can successfully and efficiently reconstruct loci of interest from several datasets. It has
several advantages over the whole genome de novo assembly such as the running time and
required computing resources. The contigs assembled by SRAssembler can also be used to
evaluate the quality of whole genome assemblies. The source code is available at
http://grinch6.gdcb.iastate.edu/~hchou/SRAssembler/.

¹Department of Genetics, Development & Cell Biology, Iowa State University, Ames, Iowa
50011, USA
²Department of Biology, Indiana University, Bloomington, Indiana 47405
³School of Informatics and Computing, Indiana University, Bloomington, Indiana 47405

*Corresponding author
Rationale

Next generation sequencing (NGS) approaches have become one of the most widely used biotechnology tools [74]. Applications of NGS include the generation of detailed maps of genetic variation [31, 32, 75], DNA methylation [33, 34], and transcription factor binding sites (Chip-Seq) [36, 37]. NGS technologies also provide opportunities to de novo sequence genomes of non-model species at very low costs. Because NGS relies on extensive sequence coverage with small reads, accurate assembly of the reads to large contigs, scaffolds, and pseudochromosomes is an intrinsic part of the approach. A large number of NGS assembly tools have been developed for this purpose. Based on de Bruijn graphs [76], such programs have been shown to effectively handle millions of short reads, including, for example, ABySS [24], ALLPATH [25], and SOAPdenovo [26].

Current challenges in assembly focus on increasing average contig size, usually measured as N50 size (N50 is the median length of contigs. People usually use it to evaluate the quality of the assembly) [77], and reducing error rates. Some strategies have been introduced to deal with these problems, such as gene-boosted assembly [78] and homology-guided assembly [79]. The assemblies based on highly related genomes have been shown to produce better results by incorporating homologous sequence information. Another challenge with de novo assemblers is that assembling the massive amounts of data is still a very difficult technical task. For eukaryotic genomes, de novo assembly typically requires computational resources with very large memory and fast processors. Even with such extensive computational resources, an assembly may take several days to complete even for a single run, and, depending on the complexity of the input data, it may not finish at all. Furthermore, if the resulting assembly is not satisfactory, parameter adjustments and
subsequent runs and comparative evaluation of different draft assemblies are typically required. These challenges must inevitably be overcome ultimately to get a reliable whole-genome assembly. For example, one of the most important parameter is the \( k \)-mer value, which is the overlap length in the de Bruijn graphs. In most cases, we do not know how to assign the best \( k \) value. The only solution seems that we need to try different \( k \) value and pick the one with the best N50 size.

Whole-genome assembly is not necessarily the immediate nor the only goal of genome-wide NGS approaches. Because of the cost-effectiveness of NGS technologies, a research group may well choose genome-wide NGS for a species even if they are interested in only a subset of the species’ genes, for example, homologs of genes already identified in other species as involved in a specific biochemical pathways or cellular structures. In those cases, it becomes desirable to restrict the assembly to those genic regions only; that is, instead of assembling the entire genome, we want to assemble the reads which correspond to annotated homologous genes of interest only. Adopting this strategy enables us to focus the assembly on specific regions, drastically reducing the required resources and running time. In pursuit of this goal we have developed the program SRAssembler (Selective and Recursive local Assembler). SRAssembler uses protein or cDNA sequences from a related species as query input to find and assemble NGS reads from a novel sequencing project for a species of interest. Potentially homologous reads serve as queries for the next recursive round of assembling local contigs, representing essentially an “in-silico” chromosome walking strategy as originally developed for mining the now outdated NCBI trace archive with the Tracemblerr program [80]. User can specify the success criteria that determine the break
condition for the recursion. At the last stage, the original queries are spliced aligned to the
draft contigs and the potential gene structures are identified.

SRAssembler is implemented as a C++ program that relies on a number of external
programs for string matching, assembly, and spliced alignment. Default pre-requisites are
Vmatch [81, 82], SOAPdenovo [83], Bowtie [15], and GenomeThreader [84]. It supports
Message Passing Interface (MPI) [85] parallel computing. The reads data can be split into
several parts so that the local alignment can be executed at the same time, therefore speeding
up the running time. In each round of the recursion, different values of k-mer, the overlap
length parameter in the de Buijn graph analyses, are tested simultaneously. Since the k-mer
parameter significantly affects the quality of assembly, different k-mer should be tested in
order to find the best one. The criterion to pick the best assembly is by evaluating the length
of contigs produced. The assembly that produces the longest contig is considered the best
assembly.

Here we demonstrate SRAssembler can successfully assemble the Arabidopsis
thaliana loci by Oryza sativa (rice) protein sequences. In this case, it is seen that
SRAssembler can distinguish and recover all the related paralogous genes even their
similarity is very high. We also show the SRAssembler could serve as an ideal tool to
evaluate the quality of whole-genome assemble by testing the core eukaryotic genes.
Results and discussion

Assembly of homologous loci from simulated data

As a first test of the applicability of the SRAssembler strategy to construct local assemblies of NGS reads that would encode putative homologs of query protein probes, the program was tested on simulated read data from Arabidopsis chromosome 1 in pursuit of assembly of (known) homologs of two representative rice proteins. The two rice genes were accessions OS01G18860.2 (S-adenosylmethionine synthetase, putative, expressed) and OS06G04560.1 (armadillo/beta-catenin repeat family protein, putative, expressed), with unique Arabidopsis homologs AT1G02500.1 (SAM1) and AT1G01950.1 (ARK2), respectively. Figure 2.1 depicts the gene structure of the Arabidopsis genes and a spliced alignment of the rice proteins onto the Arabidopsis genome sequence produced by GenomeThreader [84]. AT1G02500.1 is a relatively short gene structure consisting of only two exons, with the upstream exon entirely 5’-UTR (and therefore not covered by the rice protein spliced alignment). AT1G01950.1 is a 19-exon gene structure spanning 5,241bp and represents a relatively long gene model in Arabidopsis. The detailed exon alignment scores are shown in Table S1. There are 4 exons whose alignment scores are lower than 0.5. (The score range is from 0 to 1). Moreover, the average exon length of this gene is 132bp, which is relatively short compared to the average exon length 185bp in rice and 155bp in Arabidopsis [86]. The short average exon length as well as a poorly conserved internal region (circled in Figure 2.1) should make local assembly of this locus based on the initial protein query challenging.
As a measure of success of SRAssembler, the longest assembled contig was matched against the known Arabidopsis locus. Results are shown in Table 2.1. Depending on the simulated read coverage, SRAssembler assembled partial or complete loci, including the UTR and divergent internal regions. At 25X chromosome coverage, both loci were assembled completely and correctly.

Figure 2.2 demonstrates the recursive sampling strategy employed in SRAssembler by example of the AT1G01950.1 assembly. For each recursion round, all reads identified thus far as potentially part of a homologous locus were mapped to the final contig (using Bowtie [15]) and visualized with the Integrative Genomics Viewer (IGV) [87]. It is seen that in the initial round, all reads are locally aligned to the exons. Because the simulation was based on paired-end reads, both ends of the reads will be included as long as either one of them was mapped to an exon part, which makes the in silico chromosome walking in part more akin to “chromosome jumping”. Note that, expectedly, no reads are aligned to the long central intron in the middle in the first round. In the round 3, we can see more and more reads filling out adjacent regions. In the round 5, whole region is covered by reads. These reads are assembled into a complete contig with 100% identity compared to the TAIR10 genomic sequence [88].
Figure 2.1 The spliced alignment of OS01G18860.2 and OS1G04560.1.

We used GenomeThreader to align these two rice genes, accessions OS01G18860.2 and OS1G04560.1 (black) to Arabidopsis genome. We also show the gene structure of their homologous genes AT1G02500.1 and AT1G01950.1 (blue). The unconserved regions are highlighted by red circles, including the 3’ UTR, 5’ UTR and three exons in AT1G01950.1.
Figure 2.2 Chromosome walking of gene AT1G01950.1.

We mapped the reads of each round back to the final contig. The blue bars are the exons predicted by GenomeThreader. Mapped reads are shown as either red or light blue arrows, representing forward and reverse orientation respectively. In the initial round, all exons are locally aligned by reads. Because we used paired-end reads, some introns are also mapped if either end is mapped to the exon part. As the iteration goes, reads are mapped toward the long intron part in the middle and the both ends of the target locus.
The simulation results of two Arabidopsis genes, accessions AT1G01950.1 and AT1G02500.1. We simulated reads data with coverage from 10X to 25X. We used BLASTN to align our contigs to the gene loci. The target loci length is 5,240 and 2,186 respectively. The aligned length refers to the length in the target loci successfully covered by SRAssembler contigs. The identity is the percentage of identical nucleotides in the aligned regions. The results show that both genes are correctly assembled when the read coverage is 25X.

<table>
<thead>
<tr>
<th>Reads Pairs</th>
<th>Reads Coverage</th>
<th>AT1G01950.1 (Locus length: 5,241)</th>
<th>AT1G02500.1 (Locus length: 2,187)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Aligned Length</td>
<td>Aligned Percentage</td>
</tr>
<tr>
<td>2,173,405</td>
<td>10X</td>
<td>1,681</td>
<td>32%</td>
</tr>
<tr>
<td>3,260,107</td>
<td>15X</td>
<td>1,879</td>
<td>36%</td>
</tr>
<tr>
<td>4,346,810</td>
<td>20X</td>
<td>5,241</td>
<td>100%</td>
</tr>
<tr>
<td>5,433,512</td>
<td>25X</td>
<td>5,241</td>
<td>100%</td>
</tr>
</tbody>
</table>

Assembly of homologous loci from real data

In real experiments, NGS reads are typically not uniformly distributed over the genome sequence. To test SRAssembler performance over a wider range of potential applications with varying local read coverage as well as varying query to assembled gene product similarity, we selected loci from the Arabidopsis chromosome 1 segment from 200,000 to 1,000,000. The PlantGDB AtGDB rice homologs track [89] indicated 11 loci in that range that are at least 20kb apart and have rice homologs. Results of SRAssembler with the 11 rice proteins as query are shown in Table 2.2. Typically SRAssembler assembles many contigs including those we are not interested in. The GenomeThreader [84] can normally reduce the number of contigs to less than 10. In most cases, they are from partial or complete duplication events. In this test, we used BLASTN to align these contigs to Arabidopsis genome. The results in Table 2.2 show the contigs mapped to the regions of
interests. For most of the loci, SRAssembler successfully assembled contigs with high identity and coverage scores. The evolutionary distance information is obtained by GreenPhyl website [90]. The most distant genes are AT1G02830.1 (Ribosomal L22e protein family) and OS3G22340.1 (60S ribosomal protein L22-2, putative, expressed), whose evolutionary distance is 0.36. But this locus can be assembled without any problems. The actual evolutionary distance was computed using PROTDIST from package PHYLIP [91]. For some genes we did get the very good aligned percentage for whole loci, but we got very good results if we consider the coding region only (See column 8 in Table 2.2). For example, the aligned percentage of whole locus for gene AT1G02500.1 is only 52% for the entire locus including UTR regions, but it was 96% for the coding region.
Table 2.2 - The results of 11 Arabidopsis genes.

This table shows the results of the local assembly of 11 genes. The locus start and end are the coordinates of the loci of the Arabidopsis genes. We used BLATN to align SRAssembler contigs to the genomic sequences. The contig start and end indicate the regions mapped by SRAssembler contigs. The aligned percentage refers to the proportion of target regions can be aligned by SRAssembler contigs. We can see we can get better aligned percentage if we consider the coding regions only. The evolutionary distance also provides the information how these two homologous genes are related.

<table>
<thead>
<tr>
<th>Arabidopsis genes</th>
<th>Oryza Sativa genes</th>
<th>Locus Start</th>
<th>Locus End</th>
<th>Contig Start</th>
<th>Contig End</th>
<th>Aligned Percentage - Whole Locus</th>
<th>Aligned Percentage - Coding Region</th>
<th>Evolutionary Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT1G01560.1</td>
<td>OS08G06060.1</td>
<td>202,136</td>
<td>204,335</td>
<td>202,531</td>
<td>204,754</td>
<td>82%</td>
<td>86%</td>
<td>0.23</td>
</tr>
<tr>
<td>AT1G01750.1</td>
<td>OS02G44470.1</td>
<td>275,366</td>
<td>276,310</td>
<td>274,587</td>
<td>277,586</td>
<td>100%</td>
<td>100%</td>
<td>0.21</td>
</tr>
<tr>
<td>AT1G01820.1</td>
<td>OS06G03660.1</td>
<td>296,001</td>
<td>298,120</td>
<td>295,028</td>
<td>298,873</td>
<td>100%</td>
<td>100%</td>
<td>0.23</td>
</tr>
<tr>
<td>AT1G01950.1</td>
<td>OS06G04560.1</td>
<td>325,379</td>
<td>330,619</td>
<td>325,292</td>
<td>331,411</td>
<td>100%</td>
<td>100%</td>
<td>0.14</td>
</tr>
<tr>
<td>AT1G02130.1</td>
<td>OS01G08450.1</td>
<td>400,035</td>
<td>401,882</td>
<td>400,358</td>
<td>402,227</td>
<td>82%</td>
<td>100%</td>
<td>0.07</td>
</tr>
<tr>
<td>AT1G02500.1</td>
<td>OS01G18860.2</td>
<td>518,251</td>
<td>520,437</td>
<td>519,028</td>
<td>520,172</td>
<td>52%</td>
<td>96%</td>
<td>0.08</td>
</tr>
<tr>
<td>AT1G02830.1</td>
<td>OS03G22340.1</td>
<td>625,145</td>
<td>625,608</td>
<td>625,022</td>
<td>627,834</td>
<td>100%</td>
<td>100%</td>
<td>0.36</td>
</tr>
<tr>
<td>AT1G03190.1</td>
<td>OS05G05260.1</td>
<td>775,527</td>
<td>780,027</td>
<td>774,628</td>
<td>779,739</td>
<td>94%</td>
<td>97%</td>
<td>0.18</td>
</tr>
<tr>
<td>AT1G03330.1</td>
<td>OS8G05850.1</td>
<td>817,983</td>
<td>819,563</td>
<td>817,634</td>
<td>820,053</td>
<td>100%</td>
<td>100%</td>
<td>0.2</td>
</tr>
<tr>
<td>AT1G03475.1</td>
<td>OS4G52130.1</td>
<td>869,051</td>
<td>871,211</td>
<td>868,725</td>
<td>871,723</td>
<td>100%</td>
<td>100%</td>
<td>0.23</td>
</tr>
<tr>
<td>AT1G03630.1</td>
<td>OS10G35370.1</td>
<td>907,642</td>
<td>909,376</td>
<td>906,858</td>
<td>909,850</td>
<td>100%</td>
<td>100%</td>
<td>0.24</td>
</tr>
</tbody>
</table>

Paralogous genes

Since gene duplication is a very common event in plant species, it is important to identify and assemble these paralogous loci. Here we also tested if SRAssembler can distinguish and assemble the paralogous genes. For Arabidopsis gene AT1G01820.1 (PEX11C), which is a member of peroxisomal biogenesis factor 11 family, we have two other paralogs located in chromosome 2 and 3 respectively. In Table S2, we summarize this gene family. This gene family information is obtained from Inparanoid web site [92]. Now
we used their rice homologous gene, OS6G03660.1 (Peroxisomal membrane protein PEX11-1, putative, expressed), to test if we can find all these three loci. The results are shown as Table 2.3. We can see SRAssembler successfully assembled all paralogous genes.

Table 2.3 - The results of assembling gene family of peroxisomal biogenesis factor.

We demonstrate SRAssemble can perfectly identify all paralogs of gene family peroxisomal biogenesis factor 11 in Arabidopsis. The three paralogs are located in chromosome 1, 2 and 3. The locus start and end are the coordinates of the loci of the Arabidopsis genes. We used BLASTN to align SRAssembler contigs to the genomic sequences. The contig start and end indicate the regions mapped by SRAssembler contigs. The identity is the percentage of identical nucleotides in the aligned regions. We can see that the target regions are completely covered by SRAssembler contigs. The evolutionary distance is the distance between Arabidopsis and the query rice genes.

<table>
<thead>
<tr>
<th>AT genes</th>
<th>OS genes</th>
<th>Locus Start</th>
<th>Locus End</th>
<th>Locus Length</th>
<th>Contig Start</th>
<th>Contig End</th>
<th>Identity</th>
<th>Aligned Percentage</th>
<th>Evolutionary Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT1G01820.1</td>
<td>OS6G03660.1</td>
<td>296,001</td>
<td>298,120</td>
<td>2,120</td>
<td>295,901</td>
<td>298,417</td>
<td>100%</td>
<td>100%</td>
<td>0.23</td>
</tr>
<tr>
<td>AT2G45740.1</td>
<td>OS6G03660.1</td>
<td>18,839,865</td>
<td>18,841,102</td>
<td>1,238</td>
<td>18,839,137</td>
<td>18,841,713</td>
<td>99%</td>
<td>100%</td>
<td>0.29</td>
</tr>
<tr>
<td>AT3G61070.1</td>
<td>OS6G03660.1</td>
<td>22,604,873</td>
<td>22,606,159</td>
<td>1,287</td>
<td>22,604,702</td>
<td>22,606,283</td>
<td>99%</td>
<td>100%</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Core eukaryotic genes

We also used SRAssembler to test core eukaryotic genes. These genes are present and conserved in a wide range of species. There are 458 highly conserved core genes identified by CEGMA (Core Eukaryotic Genes Mapping Approach) study [93]. We examined first 30 such proteins with size larger than 100 amino acids. We used Drosophila melanogaster core genes as query proteins to assemble the homologous regions of Acromyrmex echinatior (leaf cutting ants). Out of these 30 proteins, 25 contigs are identified with good alignment score and coverage in the GenomeThreader report. In the last round of SRAssembler, the k-mer
value 45 was chosen in terms of the length of contigs. Since we have very high confidence in these contigs, we can use them to evaluate the quality of whole-genome *de novo* assemblies.

We aligned the SRAssembler contigs to four whole-genome assemblies. The first one was assembled by SOAPdenovo with one 500bp library using *k*-mer value 27. Its N50 value is 1,927bp. We assembled the second assembly with the same dataset as the first one, but used *k*-mer value 45, which is determined by the last round of SRAssembler. This assembly has N50 value 2,346. For the third assembly, we used an additional 2kb mate-pair library, whose N50 is 4,326bp. The last one is the assembly submitted to NCBI whole genome shotgun sequencing project by Beijing Genomics Institute (accession no. AEVX00000000) [94], which contains 4 libraries with various insert size. Its N50 is 80,630bp. The Figure 2.3 shows the comparison of these assemblies. The aligned percentage in Figure 2.3 refers to the percentage of SRAssembler contigs aligned to whole-genome assemblies. If a SRAssembler contig can align to multiple contigs in whole-genome assemblies, we simply pick the best one. Our results indicate that quality of the contigs generated by the first assembly is not very good, because they are relatively fragmented when aligned to SRAssembler contigs. Here we got only one contig completely matched. The assembly with two libraries is highly consistent with SRAssembler contigs, showing the better quality of this assembly. We can see that the mate-pair information is very useful to merge short contigs into big ones. The last assembly, with four libraries, performs the best in our comparison. They are almost identical to the SRAssembler contigs. We also noticed that the *k*-mer value chosen in the first assembly is problematic. Since SRAssembler can test different *k* value at the same time, we can use the *k* value suggested by SRAssembler to do the whole-genome assembly as we did in the second
assembly. We can see the significant improvement when we switch $k$ value from 27 to 45. The detailed results are shown as Figure 2.3 and Table S3.

![Comparison of Assemblies](image)

**Figure 2.3 Comparison of assemblies.**

We assembled 25 contigs from 25 core eukaryotic genes. Then we aligned them to three whole-genome assemblies. We can see the AEVX00000000.1 assembly is highly consistent with SRAssembler contigs, and SOAPdenovo assembly with $k$-mer 27 is relatively fragmented. Therefore, we can use SRAssembler contigs to evaluate the quality of whole-genome assemblies.

**Running time**

Since SRAssembler assembles homologous regions directly, it can finish the assembly in very short time. Here we selected a rice gene accession OS06G03660.1 (Peroxisomal membrane protein PEX11-1, putative, expressed) and aligned it to the 30 million Arabidopsis genomic reads. To understand how parallel computing benefits the running time, we tested the SRAssembler on the TACC Ranger HPC system [95]. The memory capacity is 32GB, but SRAssembler only requires lesser than 2GB in such test run.
We tested 15 rounds (this is a typical number of rounds) with two datasets. The first one used one library, SRR073127, whose size is 32.4 million reads. The second one has two libraries. In addition to SRR073127, we added another library, SRR071796, which has 30 million reads. The read length for both libraries is 75bp. The resulting contig perfectly covers the target region and the identity is 100%. Here we only focus on the running time. We can see when we have only one core, the execution time is around 147 minutes to finish 15 rounds for two libraries. As we used more CPUs up to 16 cores, we found the execution time dropped to 45 minutes, which means we can correctly assemble a gene out of 60 million reads in 45 minutes. For single library, the running time is only around 35 minutes. The results are shown as Figure 2.4. Note that these results include the pre-processing step, which splits the reads data into smaller ones. This step can be skipped when we rerun the same dataset, which can significantly reduce the running time further.
Figure 2.4 Running time of SRAssembler.

Since SRAssembler assembles homologous regions directly, it can finish the assembly in very short time. For single core, the execution time is around 147 minutes to finish 15 rounds for two libraries. As we used more CPUs up to 16 cores, we found the execution time dropped to 45 minutes, which means we can correctly assemble a gene out of 60 million reads in 45 minutes. For single library, the running time is only around 35 minutes.
Conclusions

SRAssembler provides a new way to assemble the genes of interest directly, which is extremely efficient if people are only interested in a few genes. From the tests we have seen, SRAssembler seems to be able to assemble partial or whole the homologous region correctly. It has several advantages over the whole genome de novo assembly. First, it is much faster. For the rice test cases, with the help of parallel computing, we can find homologous regions within 40 minutes. Second, it requires less resource. For a typical search, 2GB memory is enough for most cases. Third, it is flexible. Since the SRAssembler is implemented as object-oriented framework, it is very easy to add new internal assemblers. Also, different $k$ values can be tested simultaneously. If users have a 16-core machine, 15 different $k$-mer values can be tested at the same time. Therefore, SRAssembler can quickly select the best $k$ for the whole-genome assembly as shown in the second assembly in the test case of core eukaryotic genes. From the comparison with the whole-genome denovo assembly strategy, we found SRAssembler can generate identical or better results than the global assemblies. Since SRAssembler can assemble the contigs with very short time, it could serve as a benchmark to quickly evaluate the quality of the global assemblies. For the assemblies of eukaryotic species without reference genome, we can always use SRAssembler to assemble some of 458 core eukaryotic genes first. Then a good assembly should be able to assemble them correctly no matter how well the N50 value it gets. We believe SRAssembler can serve as a good supplementary tool of whole-genome de novo assembly.

A related approach to iterative targeted and micro NGS assembly was recently introduced in the Mapsembler program [96]. Although Mapsembler also adopts the similar iterative search algorithm as did in Tracembler and SRAssembler, it is not designed to
assemble homologous loci. Mapsembler is a tool targeting specific biological events such as transposase elements or gene fusion. It cannot handle the case where the similarity between query sequences and NGS reads is not very high (<90%). In our preliminary tests, Mapsembler cannot assemble any test cases shown in this paper. It does not support protein sequences and paired-end reads as well, which are very useful for the homology search.

**Materials and methods**

**In silico chromosome walking strategy**

The basic strategy implemented in SRAssembler is depicted in Figure 2.5. Initially, NGS reads are aligned to a query sequence using the fast string matching program Vmatch [82]. If the query sequence is a protein, the matching is to all possible translations of the reads (Vmatch option -dnavsprot). Default Vmatch parameter settings in SRAssembler are initial matching length 10 for protein sequences and 30 for cDNA sequences. The default mismatches allowed are 1 for protein sequences and 2 for cDNA sequences. The matching length for recursive rounds is 30. These setting can be changed by the user. Retrieved reads from this initial matching are assembled into contigs which become the query sequences for subsequent rounds of in silico chromosome walking. By default, SRAssembler invokes SOAPdenovo for the assembly step. During the assembly step, the assembler is run multiple times with different k-mer values (the default setting is 15, 25, 35, and 45). We assume the best k-mer is determined by the length of longest contig. The contigs produced by the best k-mer will become the query sequences for the next round. The recursion is terminated as soon as one of the following criteria is met: (1) No new reads can be found; (2) A specified maximum number of iterations is reached; or (3) All contigs match or exceed a specified
maximum length. The spliced alignment program GenomeThreader [84] is used to map the original query onto all assembled contigs.

![In silico chromosome walking strategy diagram](image)

**Figure 2.5 In silico chromosome walking strategy.**

The SRAssembler first aligns query sequences to the reads data. The reads initially mapped are shown as blue pairs. Note that both pair will be included as long as either end of the read is mapped. Then these reads serve as seeds to “walk” through the chromosome. The adjacent reads are searched by this “walking” strategy (red pairs). All reads we get will be assembled as contigs, and then the predictive gene structure could be obtained by spliced alignment tools to align the original query sequences to the assembled contigs.

**Implementation**

SRAssembler is implemented in C++ and compiles with any standard C++ compiler. Input read files can be in either FASTQ or FASTA format. Although SRAssembler accepts single end reads, paired-end reads always provide much better results. SRAssembler also
supports multiple libraries. Libraries with different insert size can improve the quality of assemblies. For example, some mate-pair reads with very long insert size are very helpful to merge two contigs into a big one. The query sequences can be either protein or cDNA sequences provided in FASTA format. The SRAssembler workflow is shown in Figure 2.6. SRAssembler supports compilation under the Message Passing Interface (MPI) [85]. To realize the MPI protocol, the SRAssembler first splits the input reads file into smaller chunks, which can be aligned on different nodes. One node will serve as the master node, which sends the split read file to slave nodes and finally merges the alignment results as one file. The reads that aligned to the query sequences can either serve as new query sequences in the next round or first be assembled to contigs, depending on the parameter settings. In the latter case, very short contigs will be removed. The remaining contigs become the new query sequences for the next round. In each round, the contig size is checked. If the contig size is larger than the predefined maximum value (default value 10,000), SRAssembler will stop assembling such contigs. Because they are long enough, we do not want to waste our time to keep assembling them again. SRAssembler also remove the reads associated with these contigs, therefore improving the running time. This is done by the following steps:

1. In each round, we test if the contig length is larger than the maximum contig size. We trim the head and tail of the contigs and make their size be equal to the maximum contig size, and then copy these contigs to the candidate long contig file. Note that we do not remove them immediately, because we want to do the double check if these long contigs are correctly assembled. If such contigs are assembled again, we can confirm they are our final contigs.
2. In the next round, we align the candidate long contigs to the current assembled contig file (done by Vmatch). If matched, we move the contigs to the permanent long contig file.

3. We align current matched reads to the long contigs (done by Bowtie). If matched, those reads are removed from the reads pool.

4. Long contigs are removed from the query file of the next round.

Users can also use option –r to indicate when SRAssembler should remove the reads cannot be mapped to current contigs. For example, assigning –r option 5 will make SRAssembler do the cleaning every 5 rounds. These reads are considered as “noise” reads, which may affect the accuracy of the assembling and increase the running time.

After the recursion step is done, we use the spliced alignment tools to align the original query sequences to the final contigs. Because the length of reads is very short, some incorrect or irrelevant contigs may be assembled. To evaluate the quality of these contigs, we use the GenomeThreader to map the homologous query sequences to them. If the contigs are the homologous regions of interests, the potential gene structure should be identified by GenomeThreader. In most cases, the contigs with longest predicted open reading frame should be the ones we are looking for.
Figure 2.6 Proposed pipeline of SRAssembler.

The pipeline takes as input the DNA reads and query sequences. The query sequences could be either cDNA or protein sequences. In the preprocessing step, reads data is split into smaller ones so that we can align them in parallel. Then we use Vmatch to align reads locally. Alignment reports are merged to one file. Then we use these initial hits as seeds to do the chromosome walking by recursively mapping back to the DNA reads until either no reads found or maximum rounds reached. In each round, we assemble the cumulated hit reads by SOAPdenovo assembler. The assembled contigs are the new query sequences used to find new reads. Finally, we use GenomeThreader to align the original query sequences to the final contig file, thus obtaining the proposed gene structure.
Simulation

We simulated paired-end NGS sequencing of chromosome 1 of Arabidopsis thaliana using the wgsim program of SAMTools [97]. The number of reads N was calculated as $N = \frac{\text{length of chromosome 1} \times \text{coverage}}{\text{length of reads} \times 2}$. Parameters were set as follows: base error rate 0.02, mutation 0, and the fraction of indels 0.10. Coverage was set to 10X, 15X, 20X, or 25X. Read length was set to 70bp, and insert size to 200bp with standard deviation 50bp.

Datasets

The datasets we used for Arabidopsis thaliana assembly are two libraries from Arabidopsis 1001 Genomes Project. They were downloaded from NCBI SRA database [98, 99]. The total number of spots is 16.2 million for SRR071796; 15 million for SRR073127. Before we can use this dataset, we need to first preprocess it so that we can get higher quality assemblies. We removed low quality reads and trimmed the adapter sequences. Since such preprocessing may cause orphan reads, we shuffled reads into single file so that we can easily deal with orphan reads. We first use the shuffleSequences_fastq.pl of Velvet [23] package to make an interleaved FASTQ file. Then we used ea-utils’ fastq-mcf [100] to filter out the low quality reads and trim the adapter sequences. The resulting file is then split into paired-end reads and single reads, whose number are 30,159,924 and 832,218 respectively. As for the test case of Acromyrmex echinatior, we have three assemblies. The first one was assembled from one Illumina paired-end reads library, whose SRA accession no. is ERR034186 [101]. The insert size of this library is 500bp. The number of paired-end and single reads after preprocessing are 105,032,566 and 875,341. In addition to the library ERR034186, the
second assembly used another library, SRA accession no. ERR034188 [102], whose average insert size is 2000bp. There are 134,991,658 paired-end reads and 1,315,613 orphan reads after preprocessed. The third assembly is published by Beijing Genome Institute, which includes 4 libraries with insert size 500bp, 2kb, 5kb and 10kb [103]. The total reads count is 374,604,674.

Availability and requirements

SRAssembler source code and instructions are available at http://grinch6.gdcb.iastate.edu/~hchou/SRAssembler/.
CHAPTER 3 GENOME-WIDE SURVEY OF STEPWISE INTRON REMOVAL BY RNA-SEQ DATA

A paper to be submitted to BMC Bioinformatics

Hsien-chao Chou¹, Volker P. Brendel ²,³*

Abstract

Background

To explore how the long introns are spliced, several studies have been proposed. The first hypothesis is recursive splicing (type I RSSs), which removes the sub-fragment of the introns stepwise. The most striking feature of the recursive splicing is the juxtaposition of a 3’ acceptor site and a 5’ donor site, which results in a zero length exon. Recursive splicing has been shown an important mechanism in Drosophila. However, only very small number of recursive sites can be validated. Another less restrict type of stepwise intron removal process, called intrasplicing (type III RSSs), was also proposed, but this model is only based on bioinformatics analyses and no experimental confirmation is provided. Here we developed a pipeline called RSSFinder, which can identify the genome-wide RSSs using RNA-Seq data.

Results

Our results indicate the reverse recursive splicing (type II RSSs, from 3’ to 5’) barely occurs. The type I recursive splicing shows the distinct pattern in Drosophila. When examining the sequences logo and information content, Type I RSSs in Drosophila include very strong signal of polypyrimidine tract, which was not observed in other three species. We also compared our results with the previous validated RSSs in Drosophila. Out of 10 RSSs, 8 of them are identified by RSS finder. In mouse, we confirmed 332 type III RSSs under very
strict criteria, which is more than type I RSSs, showing different long intron splicing strategies may be adopted in mouse.

Conclusions

In summary, we have used RSSFinder to successfully identify various types of RSSs for four species. We show the different patterns from the viewpoint of subintron length and information content around the recursive sites. RSSFinder can serve as a useful tool to provide more reliable RSSs than previous prediction methods. Further detailed experimental studies of recursive splicing can be set forth based on these RSSs. The recursive splicing analyses on other species can also easily be done by using RSSFinder without any training datasets.

1 Department of Genetics, Development & Cell Biology, Iowa State University, Ames, Iowa 50011, USA

2 Department of Biology, Indiana University, Bloomington, Indiana 47405

3 School of Informatics and Computing, Indiana University, Bloomington, Indiana 47405

*Corresponding author
Background

Introns are common genomic elements in most of eukaryotic genes. The mechanism of intron removal of eukaryotic genes is a complex process of interaction of several factors. This process involves the precise identification of splice sites by associated splicing factors. However, when the length of introns is extremely long, correctly selecting the true splice sites become a challenging task. To explore how the long introns are spliced, several studies have been proposed. The first hypothesis is recursive splicing, which can remove the sub-fragment of the introns stepwise from 5’ to 3’. The most striking feature of the recursive splicing is the juxtaposition of a 3’ acceptor site and a 5’ donor site, which results in a zero length exon (Figure 3.1(A)). The 5’ sites are then regenerated after the removal the upstream sub-fragment, and this process may be repeated recursively. The recursive splicing has been confirmed in Drosophila melanogaster[3]. Their study used a simple Position-Specific Scoring Matrix (PSSM) based scoring model[104] to predict 165 recursive sites and 5 of them are validated experimentally. However, their prediction model may not be very accurate since the splice sites composition is not static and highly related to the size of introns and the flanking exons [105-107]. This prediction method has been refined by their successive study [108] and proposed 376 predicted recursive sites and 10 of them are supported by RT-PCR analysis. This improved model relies heavily on the special feature of upstream polypyrrimidine tract around the recursive sites. However, our results have shown that this feature may not be observed in other species, thus making this ab initio prediction model limit to Drosophila or invertebrates only. Indeed, most of recursive splicing studies focus on Drosophila family only, and whether this mechanism ubiquitous in other species is still unknown. Furthermore, only very small set of recursive sites can be validated by RT-PCR
analysis or mutation tests. The reliability of these prediction models cannot be tested genome wide.

Another less restrict type of stepwise intron removal process, called intrasplicing, has also been introduced by [109] in 2004. In this model, a long intron comprises a set of intraintrons, which are removed until the remaining intron is short enough to be spliced in single step (Figure 3.1(C)). However, this model is totally based on bioinformatics analysis and no experimental confirmation is provided. Marilyn K Parra, etc. demonstrated an example of intrasplicing in the first exon of protein 4.1 R gene, which may be coordinated with downstream alternative splicing.

In addition to the recursive splicing and intrasplicing model, Shepard [110] used computational methods to predict the recursive splice sites on insects and vertebrates. They found that insects have more abundant recursive splicing sites compared to their complementary strand, but their results did not show the significant difference in vertebrates even most vertebrates have longer introns. They also demonstrated that the large introns in vertebrates tend to have many repeat elements such as SINE and LINE. They postulate the large introns of vertebrates may form stem structures which may facilitate the splicing by bringing donor and acceptor splicing junction closer. Although this study does not have any experimental evidence to support their hypotheses, they brought up an interesting observation that the stepwise removal mechanism may not be able to handle extreme long introns in mammals.

Although several studies have been proposed in the past decade, most of them are based on computational prediction. Some studies used RT-PCR to test the existence of intermediates, but it is impossible to use this method to perform genome-wide survey. Here
we take advantage of the deep sequencing feature of RNA-Seq protocol. Since RNA-Seq data is so deep that many reads may be mapped on introns [111], some studies have used RNA-Seq data to analyze the splicing patterns [112, 113]. In our study, we developed a tool called RSSFinder, which identifies the recursive splicing sites and intrasplicing sites using RNA-Seq data. We used RSSFinder to confirm the existence of recursive splicing and intrasplicing by identifying the reads mapped on the junction of intermediates. Theoretically, the recursive splicing may also occur from 3’ to 5’. We call this type II recursive splicing (Figure 3.1). We search for the evidence of two types of recursive splicing as well as the intrasplicing. Here we use the term type III recursive splicing to refer to the intrasplicing. The recursive splice sites are denoted as RSSs. We used RSSFinder to investigate four species including Drosophila, mouse, rice and Arabidopsis. We found that recursive splicing seems not uncommon even in plant species, even whose intron size is known very short compared to animals.
Figure 3.1 Long intron splicing models.

(A) Type I recursive splicing. The left part of the intron is removed by recognizing the acceptor sites in the middle of the intron. Then a new 5 splice site will be regenerated. If we want to prove the existence of the recursive splicing, we need to find the reads that mapped to the junction of exon1 and the right part of the intron. (B) Type II recursive splicing. In this case, the right part of the intron is removed first. The new acceptor site is regenerated. (C) Type III recursive splicing or intrasplicing. The long intron is shortened by removal of subintrons without the use of splice sites of long introns. (D) Stem structure with loops may facilitate the splicing of long introns in vertebrates.
**Methods**

To prove the existence of the recursive splicing, we have to find the reads that are mapped to the junction of the exon and the spliced intron. We would like to explore whether recursive splicing is the dominant way to deal with long introns among various species. For this purpose, we developed a pipeline called RSSFinder, which includes several Perl scripts and a C++ program.

**Intron retrieval**

The intron datasets for Drosophila, mouse, Arabidopsis and rice are first constructed. The genome and annotation version are shown as in Table 2.1. The detailed statistics of retrieved introns are described in Table 3.4. An obvious observation is that mouse has larger number of introns and longer intron size. The plant species, on the other hand, have shorter introns. Since the size of some introns is extremely long, the median size of intron is far shorter than average size. A Perl script named IntronRetriever.pl parses the gff3 or gtf format annotation files and saves the introns and the flanking exons data into MySQL database.
Table 3.1 - Datasets

<table>
<thead>
<tr>
<th>Species</th>
<th>Project/Institute</th>
<th>Version</th>
<th>Number of Introns</th>
<th>RNA-Seq runs¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drosophila Melanogaster</td>
<td>Flybase</td>
<td>5.45</td>
<td>72,306</td>
<td>SRR352499~SRR352506, SRR043397, SRR040044, SRR061686, SRR070259, SRR074421, SRR168834, SRR029112, SRR038616, SRR042297, SRR364724, SRR414921.</td>
</tr>
<tr>
<td>Mus musculus</td>
<td>GRC</td>
<td>Build38</td>
<td>532,819</td>
<td>SRR001365, SRR006492, SRR037945, SRR037946, SRR037497, SRR037950, SRR037951, SRR037952, SRR099239.</td>
</tr>
<tr>
<td>Arabidopsis thaliana</td>
<td>TAIR</td>
<td>10</td>
<td>116,481</td>
<td>SRR013417, SRR013418, SRR071240, SRR089777, SRR360152, SRR391051, SRR394082.</td>
</tr>
<tr>
<td>Oryza sativa</td>
<td>RGAP</td>
<td>7</td>
<td>184,635</td>
<td>SRR037717, SRR037720, SRR037737, SRR037738, SRR037739, SRR0504369, SRR0504371.</td>
</tr>
</tbody>
</table>


² Dataset can be downloaded from DNA Bank of Japan (http://trace.ddbj.nig.ac.jp/DRASearch/submission?acc=SRA047035).

Finding RSSs

Then a way to predict all potential recursive sites for each intron is required. Each RSS is treated as a pair of regular acceptor and donor splice sites. We implemented a C++ program named RSSPredictor which is based on the latest version of SplicePredictor [69, 114], which employs the Bayesian Markov model to predict splice sites including non-canonical sites. For type I and II RSSs, we scan all intron sequences and identify all qualified acceptor sites immediately followed by a donor site. For type III RSSs, we search for all qualified donor sites and pair them with acceptor sites within specified range. The sites whose posterior probability and Bayes factor [115] are larger than specified cutoff are considered as our candidate RSSs. For each hypothetic RSS, we construct the pseudo intermediates by ligating upstream exons to regenerated 5 prime donor site (Figure 3.2(A)) for type I RSS, regenerated 3 prime RSS to downstream exon for type II RSS (Figure 3.2(B)), or upstream and downstream of subintrons for type III RSS (Figure 3.2(C)).
The Bowtie [15] indexes of these pseudo intermediates are then built using Bowtie-build program. Then RNA-Seq data are aligned to these sequences. RSSFinder can take multiple RNA-Seq libraries as input. Here we used 19 Illumina runs for Drosophila, 9 runs for mouse, 7 runs for Arabidopsis and 7 runs for rice (Table 3.1). We first used Bowtie to map all the RNA-Seq reads to the reference genomes. The reads that are initially unable to map to the reference genome are the potential reads mapped to the junctions of recursive sites. Therefore we again used Bowtie to align these unmapped reads to the pseudo intermediates according to the following rules (they are the adjustable parameters in RSSFinder): (1) Reads must span the junction at least 12bp. (2) If the shorter part of the read is less than 18bp, the number of mismatches allowed is 1. Total 2 mismatches are allowed for the whole read. (Figure 3.2(D)). All confirmed RSSs for each RNA-Seq library are then combined and the duplicates are removed. We also filter out the RSSs associated to the known alternative splicing events. In other words, only non-exonic RSSs are considered in our study.
To confirm the three stepwise intron removal models. We first create the pseudo intermediates by concatenating intermediates by (A) ligating upstream exons to regenerated 5 prime donor site for type I RSS, (B) regenerated 3 prime RSS to downstream exon for type II RSS, or (C) upstream and downstream of subintron for type III RSS. Then we align unmapped reads to those pseudo transcripts. (D) An RSS is confirmed by RNA-Seq reads if (1) the length of shorter part of the reads mapped on the pseudo intermediates must be larger than or equal to 12bp, and (2) if the shorter part of the read is less than 18bp, the maximum mismatches allowed in this part is 1.

**Figure 3.2 Confirmation of RSSs.**
Results and discussion

We tested four model species including Drosophila, mouse, Arabidopsis and rice with simulated and real intron sequences. We then used RNA-Seq data to find the confirmed RSSs according to the rules mentioned in the method section.

Simulation

To understand whether the RSSs are the results of evolutionary pressure or simply formed by chance, we simulated the same number of introns and the flanking exons by conserving dinucleotide composition. Since the similar study has been done by Ott et al. [109] for intrasplicing, here we focus on the case where the acceptor site is immediately followed by donor site. We used relative stringent criteria to search for the RSSs with cutoff p-value (posterior probability) 0.85 and c-value (Bayes factor) 3. We found out that the predicted number of RSSs is significantly higher than the sites found in random sequences for all four species. We also noticed that in Drosophila, there is an obvious bias that recursive sites tend to happen in longer introns. But we did not find this bias in other three species. The results are shown as Figure 3.3.
Figure 3.3 Comparison of simulated and real intron sequences.

These are the results of predicted recursive sites distribution normalized by million nucleotides. We compare the number of sites with the sequences generated by first order Markov model. Among these four species, we found the number of sites are significant larger than random sequences, indicating that the recursive splicing may be a common mechanism in both animals and plants.

**Finding RSSs**

Although the simulation results suggest the recursive splicing may be a dormant way to process intron removal. We want to see if we can find evidence for recursive splicing from RNA-Seq data. For type I and II recursive splicing, instead of using higher cutoff, we want to
collect as many candidate sites as possible, thereby obtaining better sensitivity. We used very low p-value cutoff 0.5 and c-value 1. On the other hand, since the number of the paring between donor and acceptor sites for type III RSSs is explosive, we have to use very stringent criteria or the number of predicted sites is extremely high. Here we used p-value 0.97 and c-value 7 for mouse and Drosophila; p-value 0.9 and c-value 5 for Arabidopsis and rice. For mouse and Drosophila, the criterion of the range of subintron length is from 1,000bp to 10,000bp, and we used smaller range for Arabidopsis and rice, which is from 300 to 2,000. The detail results of candidate sites are shown in Table 3.2.

<table>
<thead>
<tr>
<th>Species</th>
<th>Type I</th>
<th></th>
<th>Type II</th>
<th></th>
<th>Type III</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cutoff</td>
<td># of sites</td>
<td>Cutoff</td>
<td># of sites</td>
<td>Cutoff</td>
<td># of sites</td>
</tr>
<tr>
<td></td>
<td>p-val</td>
<td>c-val</td>
<td></td>
<td>p-val</td>
<td>c-val</td>
<td></td>
</tr>
<tr>
<td>Drosophila</td>
<td>0.5</td>
<td>1</td>
<td>5,124</td>
<td>0.5</td>
<td>1</td>
<td>9,397</td>
</tr>
<tr>
<td>Mus musculus</td>
<td>0.5</td>
<td>1</td>
<td>397,835</td>
<td>0.5</td>
<td>1</td>
<td>242,935</td>
</tr>
<tr>
<td>Arabidopsis thaliana</td>
<td>0.5</td>
<td>1</td>
<td>1,193</td>
<td>0.5</td>
<td>1</td>
<td>1,249</td>
</tr>
<tr>
<td>Oryza sativa</td>
<td>0.5</td>
<td>1</td>
<td>8,912</td>
<td>0.5</td>
<td>1</td>
<td>10,285</td>
</tr>
</tbody>
</table>

Then we mapped RNA-Seq reads to these candidate sites. The RSSs are confirmed by these RNA-Seq reads. For type I and type II RSSs, the RSSs are defined by the location of RSSs and their upstream (downstream) exons. For type III RSSs, the RSSs are defined by the absolute position of subintron. Here we do not count the duplicate RSSs if they are shared by many genes. Our results indicate that type II RSS are barely observed, which is consistent with the results of Burnette’s study [3]. This observation may result from the co-transcriptional pre-mRNA splicing order, which has been shown the introns close to 5’ end tend to be spliced first [112]. We also observed some intrasplicing events even under such
stringent condition. We expect we can find more type III RSSs if we lower the cutoff and the range constraint. Except for Arabidopsis, we got hundreds of type I RSSs. The results are shown as Table 3.3. Note that the number of RSSs highly depends on the number of mapped reads required. For example, if the required number of reads is larger or equal to 10, the number of type I RSSs in mouse drops from 140 to 6. This observation shows that most RNA-Seq experiments are not designed to capture the splicing intermediates. Our datasets are all public, and they are mainly used to study mature mRNA. People may design some special experiments such as sequencing chromatin associated mRNAs if the targets is to search for the recursive splicing sites.

Table 3.3 - Confirmed sites.

<table>
<thead>
<tr>
<th>Species</th>
<th>Minimum number of mapped reads &gt;= 1</th>
<th>Minimum number of mapped reads &gt;= 3</th>
<th>Minimum number of mapped reads &gt;= 10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Type I</td>
<td>Type II</td>
<td>Type III</td>
</tr>
<tr>
<td>Drosophila</td>
<td>306</td>
<td>0</td>
<td>80</td>
</tr>
<tr>
<td>Mus musculus</td>
<td>140</td>
<td>1</td>
<td>332</td>
</tr>
<tr>
<td>Arabidopsis thaliana</td>
<td>48</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>Oryza sativa</td>
<td>243</td>
<td>3</td>
<td>225</td>
</tr>
</tbody>
</table>

Sequence content

Since we almost cannot find type II RSSs and type III RSSs are not representative due to the strict constraint, here we will focus on type I RSSs. In fact, the events of type I RSSs are also intensively studied in the past 15 years [3, 108, 110, 116, 117]. After obtaining all type I RSSs, we want to know whether the motifs around the RSSs are similar to regular
splice sites. The easiest way is to look at the sequences logo [118] and examine the information content [119]. The sequence logos for four species are shown in Figure 3.4. We can easily notice that in Drosophila, a strong polypyrimidine tract is observed. Similar but not obvious signal also appears in Arabidopsis. This feature may be not representative because only 48 RSSs are identified. We do not see such signal in other two species. We can further take a look at the information content plot for details (Figure 3.5). The information content for DNA sequences is obtained by:

\[ I_i = 2 + \sum_{N \in \{A,C,G,T\}} p_{iN} \log(p_{iN}) \]

Where \( i \) is the index of sequence, and \( p_{iN} \) is the frequency of nucleotide \( N \) in position \( i \).

In Drosophila, the position -5, -6, -9 and -11 have strong preference for T for both type I and type III RSSs. Our results are very close to the conclusion in [108]. Their enhanced RSS predictor shows the same strong polypyrimidine tract signal. We did not notice obvious difference from the regular sites for the donor sites.
Figure 3.4 The sequence logo around the RSSs in four species.

The top, middle and bottom figures are the sequence logos for type I&II RSSs, type III RSSs and regular sites. The range of the sequences is from 20 bases upstream to 10 bases downstream. The sequences of regular and type III site are synthesized by juxtaposition of the 3’ acceptor site and the 5’ donor site for each intron.
Figure 3.5 Information content plots.
The three data series represent type I&II RSSs (blue), type III RSSs (red) and regular sites (green) respectively. These plots are generated for 30 bases upstream and 20 bases downstream of RSSs.

**Length distribution**

Recursive splicing has been considered as one of important mechanism of large intron splicing. Although several simulation studies have confirmed that recursive splicing tend to happen in large introns, we want to examine length distribution of introns containing RSSs. Here we only focus on type I RSSs because we have imposed length constraint in finding potential the intrasplcing sites. We also show the subintron length (or RSS distance) distribution of type I RSSs. The subintron of RSSs is the length of between RSSs. If there are
many RSSs in one intron, the subintron length is the distance between adjacent recursive sites. In other words, we assume all subintrons associated with the upstream recursive splicing events are removed sequentially. The results are shown as Figure 3.6. We can see that except for Drosophila, the recursive splicing events does not have significant bias toward longer introns in all other three species. Our results are consistent with our simulation tests. They also support the conclusion of the Shepard’s bioinformatics study [110] that type I recursive splicing is not the main mechanism of long intron splicing in vertebrates. The detailed descriptive statistics are shown in Table 3.4.

<table>
<thead>
<tr>
<th>Species</th>
<th>RSSs Distance</th>
<th>Intron Length Containing RSSs</th>
<th>Intron Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>SD</td>
<td>1st Quartile</td>
</tr>
<tr>
<td>Drosophila</td>
<td>13453</td>
<td>8469</td>
<td>8025</td>
</tr>
<tr>
<td>Mus musculus</td>
<td>5828</td>
<td>8418</td>
<td>753</td>
</tr>
<tr>
<td>Arabidopsis thaliana</td>
<td>290</td>
<td>175</td>
<td>148</td>
</tr>
<tr>
<td>Oryza sativa</td>
<td>672</td>
<td>916</td>
<td>202</td>
</tr>
</tbody>
</table>

Table 3.4 - Descriptive statistics for length distribution.
Figure 3.6 Length distribution.
The top, middle and bottom subplots are subintron length, intron length containing RSSs and whole genome intron length distribution.

Confirmation of experimentally validated sites

In the previous work of type I recursive splicing studies, several sites are experimentally validated in Drosophila by (1) RT-PCR of the pre-mRNA intermediates; (2) RT-PCR of the recursive splicing lariats of each subintrons; (3) mutational analysis of regenerated 5’ splice sites [3, 108]. In Burnette’s study [3], they confirmed 3 non-exonic type I RSSs and 7 non-exonic type I RSSs are validated in their successive study [108]. Out of
these total 10 RSSs, 8 of them are identified by RSS finder (Table 3.5), indicating that the RSSFinder provides an effective way to detect the RSSs.

### Table 3.5 - The experimental validated sites found by RSSFinder.

<table>
<thead>
<tr>
<th>Gene Symbol</th>
<th>RSS Sequence</th>
<th>Intron Length</th>
<th>RSS Position</th>
<th>Found in RSSFinder</th>
<th>Acceptor p-value</th>
<th>Acceptor c-value</th>
<th>Donor p-value</th>
<th>Donor c-value</th>
<th>Read Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>kus*</td>
<td>CATATTTTATTTTTTTTCAGGTAATT</td>
<td>31536</td>
<td>16001</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>osp*</td>
<td>TCTCTCTGTTCTCTCTCTACTTTTGACAGGTAGTT</td>
<td>52306</td>
<td>10975</td>
<td>Yes</td>
<td>0.96</td>
<td>7.34</td>
<td>0.99</td>
<td>15.02</td>
<td>4</td>
</tr>
<tr>
<td>osp*</td>
<td>CTTGTCTCTTTATCTCATTTGCAAGGTAAAGC</td>
<td>52306</td>
<td>32097</td>
<td>Yes</td>
<td>0.99</td>
<td>10.61</td>
<td>0.98</td>
<td>8.49</td>
<td>13</td>
</tr>
<tr>
<td>Antp</td>
<td>TTGTATGGCTGCCCTTTTCCACTTTTAAGTT</td>
<td>27535</td>
<td>14886</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HopSOB</td>
<td>TTTACATGTCATTTTTATTTTTCAAGGTAAGT</td>
<td>48486</td>
<td>21935</td>
<td>Yes</td>
<td>0.99</td>
<td>12.17</td>
<td>0.99</td>
<td>15.90</td>
<td>8</td>
</tr>
<tr>
<td>Fas2</td>
<td>ATTGCCCTTTTTCTTCCCCACTTTTCAAGGTAGTT</td>
<td>48434</td>
<td>12458</td>
<td>Yes</td>
<td>0.87</td>
<td>4.49</td>
<td>0.99</td>
<td>23.88</td>
<td>1</td>
</tr>
<tr>
<td>Fas2</td>
<td>TCTCTCTTTTGTCTCTCTCTACTTTTGACAGGTAGTT</td>
<td>48434</td>
<td>21490</td>
<td>Yes</td>
<td>0.99</td>
<td>24.8</td>
<td>0.99</td>
<td>12.41</td>
<td>3</td>
</tr>
<tr>
<td>pyd</td>
<td>TTATACCTCCCTTTTCTTTCTTGGCAAGGTAGTT</td>
<td>21946</td>
<td>12650</td>
<td>Yes</td>
<td>0.99</td>
<td>11.82</td>
<td>0.99</td>
<td>10.94</td>
<td>20</td>
</tr>
<tr>
<td>Rapgap1</td>
<td>TTTCCCTTTTTTTTCTTTTGGCAAGGTAGTT</td>
<td>27676</td>
<td>13202</td>
<td>Yes</td>
<td>0.99</td>
<td>13.32</td>
<td>0.93</td>
<td>6.02</td>
<td>4</td>
</tr>
<tr>
<td>αMan-β</td>
<td>TTCCGATTTTTTCCCTTCTTTTGGCAAGGTAGTT</td>
<td>7257</td>
<td>1598</td>
<td>Yes</td>
<td>0.99</td>
<td>18.82</td>
<td>0.99</td>
<td>14.10</td>
<td>1</td>
</tr>
</tbody>
</table>


1: In Papasaikas’ study, he listed this site in gene ds, but the RSS sequence is the same.

2: In Papasaikas’ study, the intron size and position are 44,347 and 38,716, but the RSS sequence is the same.

### Discussion

In this study, we have developed a tool, RSSFinder, to search for the recursive sites using RNA-Seq reads data. Previous studies focus on using computational approaches to predict RSSs. However, such ab initio methods may return many false positive sites especially for the case of intrasplcing. We used RNA-Seq data and identified total 737 type I RSSs, 5 type II RSSs and 653 type III RSSs. The total number of RSSs RSSFinder obtains is highly depending on the number of RNA-Seq libraries used. Since RSSFinder supports multiple RNA-Seq libraries, users may expect higher RSSs discovered when more libraries are included. The sample methods and depth of RNA-Seq libraries are also very crucial. Because the mRNA intermediates are rapidly spliced and degraded, RSSs may be very difficult to identify if the reads coverage is too low. In addition, we should use total RNA
instead of poly(A) mRNAs because most poly(A) mRNAs are mature mRNAs, where little RSSs evidence could be found. Ideally, the RNA-Seq libraries targeting on the pre-mRNA directly may produce most RSSs in that many pre-mRNA intermediates can be preserved [112].

When we further analyzed the characteristics of RSSs discovered, the most impressive feature is the enhanced acceptor signal in Drosophila. We know the splice sites can be recognized either by exon definition [120] or intron definition. Some studies have suggested splice sites of short introns can be paired by intron definition [121, 122]. But for longer introns, paring donor and acceptor sites can be very difficult, and the exon definition may benefit the splice sites recognition [51]. For the type I and II RSSs, the recursive sites can be regarded as an exon with zero length. Since exon recognition is highly associated with the splice site signals and the exon length [123], the length of internal exon are restricted to specific range [51]. Therefore, the zero length exon makes this process extremely difficult.

Another splice site consensus information study also indicates that the information content increases as the length of introns gets longer [106]. Longer and stronger pyrimidine tract may help the acceptor sites recognition in very long introns. In other words, we can expect strong polypyrimidine tract signals when intron length is very long and the downstream exon length is very short. Our results in Drosophila support this conclusion. Another computational study also discovered this interesting feature in type I recursive splicing [108]. Figure 3.7 further demonstrates this observation. The type I RSSs in Drosophila are divided into two groups: subintron length > 7,000bp and < 7,000bp. We can easily see the long subintron group has stronger polypyrimidine tract signals. However, we do not observe such enhanced acceptor site signal in mouse. Although the intron size in mouse is much larger than that in
Drosophila, the median of subintron length is relatively shorter. The median subintron length in mouse is only 2,230bp, compared to 12,761bp in Drosophila, which means many large introns in mouse are not spliced by type I recursive splicing. Two other ways may contribute to the splicing of large introns in mouse. The first one is the secondary hairpins with large loops structures which may facilitate splicing by shortening the distance between donor and acceptor sites [110]. Such secondary structures are formed by a set of repeats which are largely observed in vertebrates. The second hypothesis is the type III recursive splicing may be an important mechanism for the large intron splicing. In mouse, under very stringent criteria, we still confirmed 332 type III RSSs (even though the read coverage is very low), which is even more than type I RSSs, but we did not find such abundant type III RSSs in other three species. Because the number of type III RSSs is obviously underrepresented, this type of recursive splicing may play an important role in large intron splicing in mammals.
Figure 3.7 Sequence logo for different subintron length.
The information content of type I RSSs in Drosophila is associated to the subintron length. 
(A) The RSSs with subintron length > 7,000bp. 
(B) The RSSs with subintron length < 7,000bp. The group with longer subintrons has stronger splicing signals.

Our results also suggest that the ordering and the paring of recursive donor and acceptor sites in large introns may be very complicated. The events of type I and type III recursive splicing may be intertwined and some RSSs may be shared by both type I and III recursive splicing. Figure 3.8 shows two examples of recursive splicing events occurred in the same intron in Drosophila. For example, in Figure Figure 3.8(A), the intron could be spliced sequentially in the order of RSS B, D, E and F. But type III splicing event may happen first, and then the shortened intron could be spliced with type I splicing from RSS D, E to F. In Figure 3.8(B), the splice site D serves as both type I RSS and type III RSS. The original donor site may compete with RSS B, therefore inducing different splicing pathways.
Theoretically, some RSSs may be skipped. For example, RSS E and F in Figure 3.8(A) are very close so that the recursive splicing may be spliced directly from D to F.

![Diagram of A2bp1](image)

![Diagram of dnc](image)

**Figure 3.8 Type I and type III RSSs in the same intron.**

This figure shows the composite recursive splicing events in one long intron. The red line denote the type I recursive splicing and green line is intrasplicing event. (A) The Drosophila gene A2bp1. The length of this intron is 61,565. The positions of splice sites are: A (11,745), B (13,047), C (19,012), D (28,266), E (41,434) and F (47,657). (B) The Drosophila gene dnc. The site D is shared by both events. The positions are: A (7,334), B (34,472), C (39,144) and D (46915).

Alternative splicing has been shown a very important mechanism for transcriptome diversity [124, 125]. The RSSs discovered in our study could also be true splice sites, which have not been annotated yet. For example, the acceptor component of type I RSSs may also be an alternative 3’ acceptor sites. The type III RSSs can be regarded as regular splice sites with two skipped upstream and downstream exons. In our model, the RSSs and alternative splice sites cannot be distinguished directly. In other words, the RNA-Seq sequences we found may actually be the evidence of newly discovered isoforms. To find out the true nature of these identified RSSs, further experimental work is needed. For example, the RT-PCR of
lariats analysis can used to confirm the mechanism of type I recursive splicing if the downstream lariat of recursive sites is detected. The lariat is the sequence where recursive 5’ site is ligated to the branch point of the next recursive splice site. Another experiment could be useful is the mutational analysis. The mutation of RSSs may block the normal intron removal pathway, therefore producing transcripts with retained intron. Moreover, we can examine the RNA-Seq reads sampled from cytoplasmic polyadenylated RNA or polysomal RNA libraries. If the isoforms can be found in these libraries, splice sites identified in our model are regular splice sites instead of RSSs.

**Conclusions**

In summary, we have used RSSFinder to successfully identify various types of RSSs for four species. We show the different patterns from the viewpoint of subintron length and information content around the recursive sites. 8 out of 10 experimental validated sites are discovered in our results. We believe RSSFinder can serve as a useful tool to provide more reliable RSSs than previous prediction methods. Further detailed experimental studies of recursive splicing can be set forth based on these RSSs. The recursive splicing analysis on other species can also easily be done by using RSSFinder without any training datasets.

**Availability and requirements**

RSSFinder source code and instructions are available at http://grinch6.gdcb.iastate.edu/~hchou/RSSFinder/.
CHAPTER 4 SPICING INTERMEDIATES ANALYSES USING RNA-SEQ DATA

A paper to be submitted to BMC Bioinformatics

Hsien-chao Chou\textsuperscript{1}, Volker P. Brendel\textsuperscript{2,3,*}

Abstract

We demonstrate the estimation of mRNA intermediate distribution using paired-end RNA-Seq data in Arabidopsis. All genes in Arabidopsis with 3 and 4 exons were examined, whose relative abundances were estimated by our models based on E-M algorithm. Our models were tested with simulated data. We observed that co-transcriptional pre-mRNA splicing is widespread in Arabidopsis and that transcription and splicing occur almost simultaneously. Our results indicates over 95\% of transcripts estimated are mature mRNA. Almost no primary transcripts are found. The relative abundance of the transcripts with the first intron removed is slightly higher than those with the second intron spliced for the genes with 3 exons. This observation is consistent with the conclusion that splicing order is from 5’ to 3’ in previous studies, but the difference is very small. The majority of transcripts have completed the splicing for the genes with 4 exons as well, but 1.7\% fewer than the genes with 3 exons.

\textsuperscript{1}Department of Genetics, Development & Cell Biology, Iowa State University, Ames, Iowa 50011, USA

\textsuperscript{2}Department of Biology, Indiana University, Bloomington, Indiana 47405

\textsuperscript{3}School of Informatics and Computing, Indiana University, Bloomington, Indiana 47405

*Corresponding author
Background

The core mechanism of mRNA processing is a complex process including transcription, capping, splicing and polyadenylation in eukaryotes. Exploring the dynamics of the mRNA processing is very difficult because these reactions do not occur independently. They have been shown coordinated by transaction [126]. For example, the phosphorylated CTD of RNA polymerase II provides a landing platform for capping and the interaction with SR proteins, which can facilitate the splicing and capping rate [127-129]. The splicing efficiency and specificity are also dependent on transcription. The recruitment levels of spliceosomes are highly correlated with the nascent transcript levels [130, 131].

One way to determine the prevalence of co-transcriptional splicing is using transcriptome sequencing data to examine the intron abundances. The mRNA libraries can either be extracted by depleting rRNA using a set of oligos that bind to rRNA or isolating poly-A RNA [132, 133]. The first method includes non-poly-A transcripts, which is ideal for the study of nascent RNAs. For example, 40% of RNA-Seq reads are mapped to intronic regions in the study of human brain and liver [134]. There are also some studies focusing on the nascent RNAs only. A technique called native elongating transcript sequencing (NET-seq) is developed in [135], which can quantify transcription with single nucleotide resolution. Another co-transcriptional splicing survey extracted the nascent RNA from Drosophila S2 cells, which were isolated and lysed with NUN buffer [112]. Their results indicate the widespread co-transcriptional pre-mRNA splicing in human and Drosophila. Such nascent RNA studies using RNA-Seq data simply mapped reads to intronic regions and quantified the abundances. For example, [112] used the quantification of intron retention ratio (reads per
base pair in introns/reads per base pair in all exons) and demonstrated that 87% of introns in Drosophila are spliced co-transcriptionally.

To further understand the detailed splicing dynamics, knowing the intronic abundances is not enough because the introns are shared by several potential RNA intermediates. **Figure 4.1** shows an RNA sequence with 2 introns. When the transcription has completed, there are still four types of mRNA transcripts: primary transcript with two introns, two intermediate transcripts with one spliced intron and the mature transcript with both introns spliced out. The estimation of relative abundances of each transcript helps us understand the detailed co-transcriptional mechanism during the transcription. To quantify the expressed transcripts or transcript isoforms, the simplest way is to count the total number of reads mapping to the related genome region. However, this count is affected by the length of the region of interest. Therefore, a quantitative normalized measure called RPKM (Reads Per Kilobase per Million of mapped reads) was introduced [71] which could be used to compare the difference of expression between genes or transcripts. To obtain more accurate estimation of the expression level, the problem that some reads may be mapped to multiple locations must be resolved. These multi-location mappable reads (MMRs) generally arise from conserved domains of paralogous gene families. Many studies simply discarded MMRs and used the reads that are mapped uniquely to the genome only [43, 136, 137], but this omission underestimates coverage of repeated regions, therefore introducing experimental bias. Some solutions are proposed to tackle this problem. For example, MMRs could be proportionally assigned to the transcripts based on the amount of uniquely mappable reads nearby [72, 138]. In [139], the authors regarded the true mappings as a latent variable and used the Expectation-Maximization (EM) algorithm to maximize the likelihood of the
expression levels for single end reads. Although the MMRs problem is a serious issue in estimation of expression levels, the introduction of paired-end reads has significantly alleviated this problem [140] by decreasing the ambiguity of reads mapping.

In this study, we want to estimate the abundances of these transcripts and intermediates using RNA-Seq data. Since we do not know the actual expression levels in real data, we first used the simulated data to evaluate the performance of our algorithms, and then examined the final abundances obtained from real data.
Figure 4.1 The calculation of the abundances of the primary, intermediate and mature transcripts.

This is an example of a gene with three exons. Transcript1 is the primary transcript. As the splicing going, two possible intermediates, transcript2 and 3 are produced. After the completion of the splicing, the mature transcript, transcript4, forms. If the reads are paired-end reads (reads from the sequencing of both ends of the fragments), we can calculate the expected abundances of the intermediates by the reads uniquely mapped on the black bar. For example, if the left end of the reads are mapped to the first intron and the right end of the read are mapped to the second intron, we can conclude that this read must come from the transcript1. Because we assume the fragments are sampled by uniform distribution, we can calculate the abundances by considering the length of the black bar.

Methods

For the convenience of the discussion, the terms intermediates and transcripts are interchangeable. We want to estimate the abundance of transcript \( t \). In the case of four transcripts, \( t = [1, 2, 3, 4] \). To distinguish the MMRs, we proposed two possible solutions: (1)
counting the reads mapped to uniquely mapped regions. (2) using E-M algorithm to cluster reads into groups using the fragment length information.

**Counting uniquely mapped reads**

This method is very straightforward. In Figure 4.1, the black bars are the regions that can be used to distinguish these four transcripts. The number of transcripts is obtained by counting the mapped reads from each transcript and normalized by its length. This value could be estimated by the following equation:

\[
c(f) = N \times f(t) \times \left( \frac{u_t}{l_t - L + 1} \right)
\]

Where \( c(f) \) is the predicted count of reads that are uniquely mapped to \( t \). \( N \) is number of total reads. \( f(t) \) is the unknown probability that a fragment comes from transcript \( t \). \( u_t \) is the number of starting points giving the uniquely mapped reads to transcript \( t \) (see Figure 4.1). \( l_t \) is the length of transcript \( t \). \( L \) is the mean length of fragments. So we can calculate \( f(t) \) from \( c(f) \) easily. Once we get \( f(t) \), we can obtain the proportions of each transcript.

The expected count of transcripts is:

\[
c(t) = N \times f(t) \times \frac{L}{l_t}
\]

The estimated proportion of transcript \( t \) is:

\[
\theta_t = \frac{c(t)}{\sum_s c(s)}
\]

Where \( \theta \) is the proportion of the transcript \( t \).
E-M algorithm

The uniquely mapped reads method requires many reads mapped on the distinguishable regions. This method has two problems: (1) the expression level must be high enough, and (2) the number of uniquely mapped reads may be very low if the length of exon2 is too short or too long. These problems may be even worse if the number of exons is more than 3. Here we introduce a more general mathematical method called E-M algorithm which can estimate the source of reads by iteratively maximizing the likelihood given the fragment length distribution. This method is based on the work of [139], which used single-end reads. Here we extend this methods and take the advantage of the assumption of that the fragment length distribution of paired-end reads is normal.

**Problem statement:** We have \( N \) fragments \( f \) and \( M \) transcripts \( t \), where \( f \) is the set of all fragments observed. \( f = [f_1, ..., f_n] \). \( t \) is the set of all transcripts. \( t = [t_1, ..., t_m] \). We want to estimate the proportion of fragments from each transcript \( \theta \), where \( \theta = [\theta_1, ..., \theta_m] \) and \( \sum_m \theta_m = 1 \). Therefore, we can denote \( P(t_m = i) = \theta_i \).

Since we do not know which transcript those fragments come from, we introduce a set of indicator latent variables \( z_{nm} \), where \( z_{nm} = 1 \) if fragment \( f_n \) comes from transcript \( t_m \) and 0 otherwise. Now we can calculate our data likelihood:

\[
P(f|\theta) = \prod_{n=1}^{N} P(f_n)
\]

We consider all possible transcripts:

\[
P(f|\theta) = \prod_{n=1}^{N} \sum_{m=1}^{M} P(f_n|t_m)\theta_m
\]
where $P(f_n|t_m)$ is the probability of fragment $n$ given the transcript $m$. And we estimate:

$$P(f_n|t_m) = P(f_n|z_{nm} = 1) = \frac{P(d_{nm})}{l_m - (2R + d_{nm}) + 1}$$

where $d_{nm}$ is the distance of two end reads of fragment $n$ when mapped on transcript $m$. We assume $d_{nm}$ follows normal distribution, so $d_{nm} \sim N(\mu, \sigma^2)$, where $\mu$ and $\sigma^2$ are the mean and variance of insert size of two end reads. Because this insert size is not a continuous variable, we should approximate this probability by considering a specified adjacent range, say 5bp. For example, if we observe an insert size is 245, then the $P(d_{nm} = 245)$ is approximated by $P(240 \leq d_{nm} \leq 250)$, which could be easily obtained by normal distribution. If fragment $n$ cannot be mapped on transcript $m$, then $P(f_n|t_m) = 0$. We also assume the fragment position in a transcript is uniformly distributed, so there are $l_m - (2R + d_{nm}) + 1$ probable positions a fragment could derive from, where $R$ is the length of reads.

The complete data likelihood with latent variables turns out to be:

$$P(f, z|\theta) = \prod_{n=1}^{N} \sum_{m=1}^{M} P(z_{nm} = 1|f_n, \theta) \frac{\theta_m P(d_{nm})}{l_m - (2R + d_{nm}) + 1}$$

In E step, we calculate the expected value of our hidden variables:

$$P(z_{nm} = 1|f_n, \theta) = \frac{P(z_{nm} = 1)P(f_n|z_{nm} = 1)}{\sum_{j=1}^{M} P(z_{nm} = 1)P(f_n|z_{nm} = 1)} = \frac{\theta_m P(d_{nm})/(l_m - (2R + d_{nm}) + 1)}{\sum_{j=1}^{M} \theta_j P(d_{nj})/(l_j - (2R + d_{nj}) + 1)}$$

Therefore, total $N \times M$ expected $z$ values are calculated. In M step, we need to maximize the Q function:

$$Q(\theta|\theta^{(old)}) = \sum_{n,m} P(z_{nm} = 1|f_n, \theta_m^{(old)}) \log(\theta_m P(f_n|z_{nm} = 1))$$
We have:

\[ \theta_m^{(new)} = \frac{N_m}{N} \]

where

\[ N_m = \sum_{n=1}^{N} P(z_{nm} = 1|f_n, \theta^{(old)}) \]

E step and M step are repeated until either the model parameter \( \theta \) or the data likelihood is converged.

Let \( \tau_m \) be the relative abundance of transcript \( t_m \) and \( L \) be the expected fragment length.

\[ \tau_m = \frac{\theta_m (l_m - L)}{\sum_{j=1}^{M} \theta_j (l_j - L)} \]

**Results and discussion**

Since we do not know the real distribution of each transcript, we performed simulations to test our methods. The simulated paired-end RNA-Seq data is error free and the fragment size follows normal distribution with mean 400bp and standard deviation 60bp. The fragment position is uniformly distributed. The number of reads is determined by the coverage specified. We found EM algorithm performs better even with low reads coverage. Then we used EM algorithm to evaluate real Arabidopsis dataset.

**Simulation – uniquely mapped reads**

We simulated an Arabidopsis gene, accession AT1G11020, with coverage 80X, 100X and 120X. This gene has three exons and two introns. The lengths of exons are 739bp, 124bp
and 554bp. The intron lengths are 303bp and 114bp. Therefore, we have four possible transcripts, and we simulated these four transcripts with the proportions described in Table 4.1. Transcript1 is the primary, unspliced mRNA. Transcript2 and transcript3 are the intermediate transcripts with the first and second intron removed respectively. Transcript4 is the mature mRNA.

**Table 4.1 - The length and proportion specified in the simulation of gene AT1G11020.**

<table>
<thead>
<tr>
<th>Transcript</th>
<th>Type</th>
<th>Length</th>
<th>Proportions of mapped reads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transcript1</td>
<td>Primary mRNA</td>
<td>1,834</td>
<td>5%</td>
</tr>
<tr>
<td>Transcript2</td>
<td>Intermediate2(the second intron removed)</td>
<td>1,531</td>
<td>20%</td>
</tr>
<tr>
<td>Transcript3</td>
<td>Intermediate2(the second intron removed)</td>
<td>1,720</td>
<td>12%</td>
</tr>
<tr>
<td>Transcript4</td>
<td>Mature mRNA</td>
<td>1,417</td>
<td>63%</td>
</tr>
</tbody>
</table>

The simulation results are shown as Table 4.2. For each simulation, we use Chi-squared test to examine if the predicted number of transcript is equal to the actual value. When the coverage is 120X, the p-value is larger than 0.05 significant level, but the other two simulations show the observed and actual number of transcripts are different, indicating the higher coverage is required in order to get better prediction.
Table 4.2 - The simulation results using uniquely mapped reads with different coverage.

<table>
<thead>
<tr>
<th></th>
<th>Coverage=80X, Number of fragments =965</th>
<th>Coverage =100X, Number of fragments =1206</th>
<th>Coverage =120X, Number of fragments =1447</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of fragments</td>
<td>Percentage</td>
<td>Number of fragments</td>
</tr>
<tr>
<td></td>
<td>Observed</td>
<td>Actual</td>
<td>Observed</td>
</tr>
<tr>
<td>Transcript1</td>
<td>36</td>
<td>48</td>
<td>3.73%</td>
</tr>
<tr>
<td>Transcript2</td>
<td>156</td>
<td>193</td>
<td>16.17%</td>
</tr>
<tr>
<td>Transcript3</td>
<td>98</td>
<td>116</td>
<td>10.16%</td>
</tr>
<tr>
<td>Transcript4</td>
<td>675</td>
<td>608</td>
<td>69.95%</td>
</tr>
</tbody>
</table>

*p-value* 0.00014925 1.33E-05 0.131065

* if p-value > 0.05, the observed value is equal to actual value.

Another problem of this method is only small fraction of reads can be used to distinguish the sources of the reads. In the case of three exons, the reads mapped to the both ends of middle exon are the informative ones. If the length of middle exon is too long, this method may not find any useful reads. For example, the length of middle exon of gene AT1G74070 is 575bp, we cannot find any informative reads even with the 120X coverage.

**Simulation - E-M algorithm**

We also used E-M algorithm to test the same gene. In this example, the initial proportion of each transcript is the same. The data likelihood is then iteratively improved until no obvious increase observed. Figure 4.2 shows the changes of log likelihood in gene AT1G11020. The total number of fragments is 965. Here only 10 rounds were needed to maximize it.
The log likelihood of AT1G11020.

The value of log likelihood increases as iteration goes. In round 10, no significant improvement is observed, and the recursion stops.

Table 4.3 shows the estimation of $\theta$ in each round. The initial values are evenly assigned. In the 10th round, the estimated $\theta$ is very close to the real value.

Table 4.3 - The estimation of $\theta$ for gene AT1G11020.

<table>
<thead>
<tr>
<th>Round</th>
<th>$\theta_1$</th>
<th>$\theta_2$</th>
<th>$\theta_3$</th>
<th>$\theta_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>2</td>
<td>0.1547</td>
<td>0.2501</td>
<td>0.1922</td>
<td>0.4030</td>
</tr>
<tr>
<td>3</td>
<td>0.1067</td>
<td>0.2386</td>
<td>0.1530</td>
<td>0.5017</td>
</tr>
<tr>
<td>4</td>
<td>0.0826</td>
<td>0.2280</td>
<td>0.1309</td>
<td>0.5585</td>
</tr>
<tr>
<td>5</td>
<td>0.0704</td>
<td>0.2205</td>
<td>0.1197</td>
<td>0.5894</td>
</tr>
<tr>
<td>6</td>
<td>0.0640</td>
<td>0.2158</td>
<td>0.1143</td>
<td>0.6058</td>
</tr>
<tr>
<td>7</td>
<td>0.0605</td>
<td>0.2131</td>
<td>0.1120</td>
<td>0.6144</td>
</tr>
<tr>
<td>8</td>
<td>0.0585</td>
<td>0.2115</td>
<td>0.1111</td>
<td>0.6189</td>
</tr>
<tr>
<td>9</td>
<td>0.0573</td>
<td>0.2107</td>
<td>0.1109</td>
<td>0.6212</td>
</tr>
<tr>
<td>10</td>
<td>0.0565</td>
<td>0.2103</td>
<td>0.1109</td>
<td>0.6223</td>
</tr>
<tr>
<td>Actual</td>
<td>0.0497</td>
<td>0.2</td>
<td>0.1202</td>
<td>0.6301</td>
</tr>
</tbody>
</table>
We also tested this gene with different coverage. In E-M algorithm, the p-value is large than 0.05 when coverage is 80X, indicating less reads required to get the quality prediction of the relative abundances (see Table 4.4).

Table 4.4 - The simulation results using E-M algorithm with different coverage.

<table>
<thead>
<tr>
<th>Transcripts</th>
<th>Coverage = 40X, Number of fragments = 482</th>
<th>Coverage = 60X, Number of fragments = 723</th>
<th>Coverage = 80X, Number of fragments = 965</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of fragments</td>
<td>Percentage</td>
<td>Number of fragments</td>
</tr>
<tr>
<td>Transcript1</td>
<td>Observed</td>
<td>Actual</td>
<td>Observed</td>
</tr>
<tr>
<td>Transcript2</td>
<td>13</td>
<td>24</td>
<td>2.10%</td>
</tr>
<tr>
<td>Transcript3</td>
<td>85</td>
<td>96</td>
<td>17.63%</td>
</tr>
<tr>
<td>Transcript4</td>
<td>42</td>
<td>58</td>
<td>8.71%</td>
</tr>
<tr>
<td></td>
<td>342</td>
<td>304</td>
<td>70.95%</td>
</tr>
<tr>
<td><em>p</em>-value</td>
<td>0.00145886</td>
<td></td>
<td>2.1E-05</td>
</tr>
</tbody>
</table>

We also tested the case of genes with 4 exons. We simulated 1,216 fragments for Arabidopsis gene AT1G11240, whose primary transcript length is 1,540bp. The p-value of the Chi-squared test is 0.25. The results are shown in Table 4.5. The estimated θ in round 11 is very close to the actual values.

Table 4.5 - The estimation of θ for gene AT1G11020.

<table>
<thead>
<tr>
<th>Round</th>
<th>θ₁</th>
<th>θ₂</th>
<th>θ₃</th>
<th>θ₄</th>
<th>θ₅</th>
<th>θ₆</th>
<th>θ₇</th>
<th>θ₈</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1250</td>
</tr>
<tr>
<td>2</td>
<td>0.0497</td>
<td>0.0590</td>
<td>0.0879</td>
<td>0.0747</td>
<td>0.0707</td>
<td>0.1770</td>
<td>0.0616</td>
<td>0.4194</td>
</tr>
<tr>
<td>3</td>
<td>0.0374</td>
<td>0.0518</td>
<td>0.0843</td>
<td>0.0528</td>
<td>0.0580</td>
<td>0.1396</td>
<td>0.0351</td>
<td>0.5410</td>
</tr>
<tr>
<td>4</td>
<td>0.0352</td>
<td>0.0555</td>
<td>0.0872</td>
<td>0.0469</td>
<td>0.0545</td>
<td>0.1092</td>
<td>0.0263</td>
<td>0.5853</td>
</tr>
<tr>
<td>5</td>
<td>0.0345</td>
<td>0.0601</td>
<td>0.0890</td>
<td>0.0448</td>
<td>0.0529</td>
<td>0.0913</td>
<td>0.0221</td>
<td>0.6054</td>
</tr>
<tr>
<td>6</td>
<td>0.0340</td>
<td>0.0640</td>
<td>0.0896</td>
<td>0.0437</td>
<td>0.0521</td>
<td>0.0811</td>
<td>0.0197</td>
<td>0.6159</td>
</tr>
<tr>
<td>7</td>
<td>0.0335</td>
<td>0.0669</td>
<td>0.0895</td>
<td>0.0431</td>
<td>0.0518</td>
<td>0.0752</td>
<td>0.0181</td>
<td>0.6218</td>
</tr>
<tr>
<td>8</td>
<td>0.0332</td>
<td>0.0690</td>
<td>0.0892</td>
<td>0.0427</td>
<td>0.0519</td>
<td>0.0718</td>
<td>0.0171</td>
<td>0.6251</td>
</tr>
<tr>
<td>9</td>
<td>0.0328</td>
<td>0.0705</td>
<td>0.0889</td>
<td>0.0424</td>
<td>0.0520</td>
<td>0.0698</td>
<td>0.0164</td>
<td>0.6271</td>
</tr>
<tr>
<td>10</td>
<td>0.0326</td>
<td>0.0716</td>
<td>0.0885</td>
<td>0.0423</td>
<td>0.0523</td>
<td>0.0686</td>
<td>0.0159</td>
<td>0.6283</td>
</tr>
<tr>
<td>11</td>
<td>0.0323</td>
<td>0.0724</td>
<td>0.0882</td>
<td>0.0422</td>
<td>0.0525</td>
<td>0.0679</td>
<td>0.0155</td>
<td>0.6290</td>
</tr>
<tr>
<td>Actual</td>
<td>0.0313</td>
<td>0.0723</td>
<td>0.1044</td>
<td>0.0419</td>
<td>0.0419</td>
<td>0.0625</td>
<td>0.0205</td>
<td>0.6250</td>
</tr>
</tbody>
</table>
**Estimation on Arabidopsis leaf data**

We used E-M algorithm to estimate Arabidopsis genes with 3 and 4 exons. Here we only consider genes without annotated alternative splicing events. The main assumption of our methods is the fragment length distribution must be normal. We examined the fragment distribution of our dataset by aligning reads to the Arabidopsis transcriptome data using Bowtie [15]. The distribution is very close to Gaussian distribution with the exception of a small fraction of fragment length less than 150bp (see Figure 4.3). We notice around 10% reads can be mapped to introns, so these very short fragments may contain the intron sequences, thus underestimating the length.

![Fragment Length Distribution](image)

**Figure 4.3** Fragment length distribution of dataset SRR391052.

RNS-Seq reads are mapped to the Arabidopsis transcriptome. We got the length distribution very close to Gaussian with mean 230bp and standard deviation 21bp.
For the case of three exons, we removed the genes with coverage lower than 70X. The genes whose intron/exon ratio is lower than 0.2 or higher than 5 were also excluded because of that when this ratio is far from 1, all transcripts will be looked very similar so that our model may not distinguish them very well. After the filtering, 134 genes were kept. Our results are shown as Figure 4.4(A). Over 95% of transcripts estimated are mature mRNA. Almost no primary transcripts are found. The relative abundance of the transcripts with the first intron removed is slightly higher than those with the second intron spliced. This observation is consistent with the conclusion that splicing order is from 5’ to 3’ in several studies [112, 134], but the difference is very small. Figure 4.4(B) demonstrates the relative abundances of the genes with 4 exons. We used the same filter criteria and got 135 genes. Still, we found the majority of transcripts have completed the splicing, but 1.7% fewer than the genes with 3 exons. This is probably because the average length of 4-exon genes is longer, and some introns may not be removed in time as the transcription is done. Except for the mature mRNA, the abundances of the rest of 7 transcripts are all smaller than 2%. No statistical difference was observed.
Figure 4.4 Results of transcript proportion estimation of dataset SRR391052.

The number of the left is the mean length of each transcript. (A) Results of genes with 3 exons. Over 95% of transcripts discovered are mature mRNA. (B) Results of gene with 4 exons.

In this study, we observed that co-transcriptional pre-mRNA splicing is widespread in Arabidopsis. Transcription and splicing occur almost simultaneously. In the previous study in Drosophila [112], 13% introns are co-transcriptionally spliced poorly. The average pre-mRNA length may contribute to this observation. For example, some Drosophila genes have very large introns [3], which may increase the process time. Since the length of pre-mRNA is shorter and the number of introns is fewer in Arabidopsis, all introns may be spliced almost at the same time. In the 4-exon case illustrated in previous section, the downstream introns may be spliced first. For example, the transcript 4, which has all intron spliced except the
second one, has 1.4% relative abundance. In the aforementioned study in Drosophila [112], they conclude that the co-transcriptional splicing is less efficient for first introns. We also observed similar pattern in the 4-exon case. The transcript6, the intermediate with first intron unspliced only, has almost highest relative abundance among all other splicing intermediates.

Note that these intermediates may also be the isoforms of a gene. In Figure 4.5, we can observe that the alternative splicing of intron retention is exactly the same as the splicing intermediate coming from the other isoform. To distinguish them, the best way is to sample the polysomal RNA, where we can confirm whether the reads are from the intermediates or isoforms. However, in our results, those intermediates have very low abundances. The expression level of alternative isoforms should be much higher than 3%.
Suppose we speculate that this gene has two possible isoforms, we notice that the intron retention version of the gene is identical to the intermediates of the other isoform. One way to distinguish them is to examine the reads sampled from cytoplasm. If we cannot find the reads that mapped on the isoform 1, we can conclude that those reads in nucleus are from the intermediate instead of the isoform.

Our model provides detailed information of relative abundances of all possible transcripts by taking advantage of paired-end information. Take the 4-exon case as an example, if we know the reads mapped to the first intron, we can further show the exact source is transcript 1, 5, 6 or 7. This model provides a basic tool to explore the real splicing dynamics. But we still have some limitations. First, if we want to predict the gene with large number of exons, the possible transcripts grow exponentially. The total number of possible transcripts is $2^n$, where the $n$ is the number of introns. Second, the accuracy may decrease if the intron length is too long unless the reads coverage is very high. In this case, the majority reads mapped on introns are identical. Only very small fraction of them is informative. Third, the RNA-Seq data has to be paired-end reads, and the fragment length distribution is the key to the success of this model. Sometimes, the mean and standard deviation information provided by sequencer is not very accurate. The users should get the real fragment length.
distribution by either mapping all reads to annotated gene sequences or doing the \textit{de novo} transcriptome assembly first and then mapping all reads back to the assembled transcripts. Note that not all RNA-Seq libraries follow the normal distribution. In that case, this model is not suitable. For example, the Figure 4.6 shows an example of the fragment length distribution of NCBI SRA dataset SRR360152. In this dataset, the fragment length distribution is obvious not normal.

![Fragment Length Distribution](image)

**Figure 4.6 Fragment length distribution of dataset SRR360152.** RNS-Seq reads are mapped to the Arabidopsis transcriptome. The length distribution is not normal.

**Conclusions**

We have demonstrated the estimation of mRNA intermediate distribution using paired-end RNA-Seq data in Arabidopsis. All genes in Arabidopsis with 3 and 4 exons are
examined, whose relative abundances are estimated by our models based on E-M algorithm, which were also tested with simulated data. Our results indicate over 95% of transcripts are mature mRNA and 99% of transcripts are co-transcriptional. Our models can serve as basis to decipher the detailed mechanism of splicing and its relationship with transcription by high-throughput sequencing. The future work may include more real data tests in other species. We also need to compare different biological replicates to see if they are correlated. The ultimate goal is to explore the splicing dynamics throughout the whole splicing process. For example, we can examine the nascent pre-mRNA reads and establish a complex splicing dynamics model by incorporating more parameters such as the nascent mRNA synthesizing, degrading and the intron splicing rates.

**Availability of supporting data**

The datasets used in this study is downloaded from NCBI SRA [52]. The accession is SRX112186. This total mRNA library was extracted from Arabidopsis thaliana leaf using Qiagen RNeasy columns, and then used Illumine mRNA-seq Library Prep Kit to select the mRNA library. The number of spots is 26,682,846. This library is paired-end with read length 76bp.
CHAPTER 5 GENERAL CONCLUSION

Summary

In this thesis, I worked on three research topics related to NGS technologies. I provide a set of tools and analyses. With high throughput sequencing, people can analyze non-model species with unprecedented high resolution. Instead of trying to assemble the whole genome, I focus on efficiently reconstructing the genomic regions related to the homologous protein or cDNA sequences. For this purpose, I developed SRAssembler, a local assembly program using the iterative chromosome walking strategy to assemble the loci of interest directly. The gene structures in these loci are also predicted by SRAssembler. The accuracy of the prediction is highly related to the similarity of query sequences. Since the assembled regions might be very short, people can use RNA-Seq evidence[43, 46] and/or other ab initio prediction programs[141-148] to further confirm the gene structure, including all possible alternative splicing isoforms. Meanwhile, the gene expression level of each isoform can be estimated as well.

These RNA-Seq reads can also be used to analyze the intron splicing model. The first mechanism I explored is the recursive splicing patterns in large introns. Splicing large introns is very challenging in terms of the splicing sites recognition because many possible pseudo splice sites may compete with the true ones. Recursive splicing may play a very critical role in large intron splicing[3]. Therefore, I developed a pipeline called RSSFinder, which can search for recursive sites confirmed by RNS-Seq data. My study provides the insight of prevalence of recursive splicing in different species. These predicted recursive sites can also be used to investigate the diseases associated to the abnormal splicing.
These RNA-Seq reads can also serve as a means to decipher the detailed mechanism of splicing and its relationship with transcript. Here I proposed mathematical models to estimate the abundances of mRNA intermediates. I evaluated the accuracy by simulated data. Then the Arabidopsis dataset was tested. My results indicate the co-transcriptional splicing is widespread in Arabidopsis thaliana.

**Recommendation for future research**

My studies can be further improved in several ways. For SRAssembler, several internal aligners and assemblers should be tested. For example, ALLPATH-LG [25] is known to outperform other de novo assemblers. It should be included in the future version of SRAssembler. The accuracy of assembly may also be improved by merging the assemblies with different k-mer values instead of picking the k producing the longest contig. Moreover, the local transcriptome assembly may also be an important feature in the future development of SRAssembler.

The application of RSSFinder can be extended to different species. For example, the survey of other insect species should provide more useful information when compared with Drosophila. The importance of intra-splicing model [109] should be further investigated with less strict searching criteria. The correlation between the secondary structure features and the position of recursive sites is also an interesting topic for the future research.

The future development of analysis of mRNA intermediates may include more real data tests in plant species. Comparison between different replicates is needed to see if they are correlated. This study is a good start to decipher the whole splicing dynamics. The splicing dynamics can be represented as an ordinary differential equation (ODE) model with
the rate parameters such as nascent transcripts synthesis, splicing, and the degradation rate. More experimental data such as time series RNA-Seq data are needed to complete this model. Also, the interaction between transcription and splicing needs to be elucidated. For example, the splicing of upstream introns may occur before the transcription of downstream exons. Since the length of intermediates is not a fixed value, estimating the transcript abundances becomes very difficult for the probability model. Furthermore, the pattern of mRNAs outside the nucleus should also be analyzed. The sequencing of mRNAs associated with polyribosomes can provide the information of transcripts ready for translation. The analysis of transcripts subjected to nonsense-mediated decay (NMD) also plays a key role to complete the whole picture of mRNA processing. An ultimately integrated mRNA processing model may be established as described in Figure 5.1.
Figure 5.1 The integrated mRNA processing model.

The gene structure is first assembled and predicted by SRAssembler. The splicing dynamics is then explored. Some isoforms are transported into cytoplasm, where some of them might be decayed through nonsense-mediated decay pathway.
REFERENCES


81. Vmatch.


86. Lu H: Application of comparative genomics for detection of genomic features and transcriptional regulatory elements. Iowa State University, Bioinformatics and Computational Biology; 2011.

87. Integrative Genomics Viewer(IGV) [http://www.broadinstitute.org/igv/]

88. The Arabidopsis Information Resource (TAIR) [http://www.arabidopsis.org/]


90. GreenPhyl [http://greenphy1.cirad.fr/v1/cgi-bin/greenphy1.cgi]
95. Texas Advanced Computing Center (TACC) The University of Texas at Austin [http://www.tacc.utexas.edu/resources/hpc]
96. Peterlongo PM, Chikhi RM: **Mapsembler, targeted and micro assembly of large NGS datasets on a desktop computer.** *BMC Bioinformatics* 2012, **13:**48.
103. EMBL-EBI ENA, accession ERP000666 [http://www.ebi.ac.uk/ena/data/view/ERP000666]
104. Goodrich JA, Schwartz ML, McClure WR: **Searching for and predicting the activity of sites for DNA binding proteins: compilation and analysis of the binding sites for Escherichia coli integration host factor (IHF).** *Nucleic acids research* 1990, **18:**4993-5000.
106. Weir M, Rice M: **Ordered partitioning reveals extended splice-site consensus information.** *Genome research* 2004, **14:**67-78.
107. Dewey CN, Rogozin IB, Koonin EV: **Compensatory relationship between splice sites and exonic splicing signals depending on the length of vertebrate introns.** *BMC genomics* 2006, **7:**311.


140. Li H, Ruan J, Durbin R: Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Research* 2008, **18**:1851-1858.


APPENDIX A. ADDITIONAL MATERIAL

Table S1: The GenomeThreader alignment report of OS06G04560.1

<table>
<thead>
<tr>
<th>Exon</th>
<th>Length</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exon1</td>
<td>19</td>
<td>0.188</td>
</tr>
<tr>
<td>Exon2</td>
<td>101</td>
<td>0.855</td>
</tr>
<tr>
<td>Exon3</td>
<td>123</td>
<td>0.916</td>
</tr>
<tr>
<td>Exon4</td>
<td>186</td>
<td>0.807</td>
</tr>
<tr>
<td>Exon5</td>
<td>222</td>
<td>0.847</td>
</tr>
<tr>
<td>Exon6</td>
<td>154</td>
<td>0.278</td>
</tr>
<tr>
<td>Exon7</td>
<td>150</td>
<td>0.919</td>
</tr>
<tr>
<td>Exon8</td>
<td>89</td>
<td>0.941</td>
</tr>
<tr>
<td>Exon9</td>
<td>204</td>
<td>0.61</td>
</tr>
<tr>
<td>Exon10</td>
<td>96</td>
<td>0.344</td>
</tr>
<tr>
<td>Exon11</td>
<td>135</td>
<td>0.338</td>
</tr>
<tr>
<td>Exon12</td>
<td>132</td>
<td>0.433</td>
</tr>
<tr>
<td>Exon13</td>
<td>133</td>
<td>0.36</td>
</tr>
<tr>
<td>Exon14</td>
<td>93</td>
<td>0.815</td>
</tr>
<tr>
<td>Exon15</td>
<td>126</td>
<td>0.942</td>
</tr>
<tr>
<td>Exon16</td>
<td>126</td>
<td>0.841</td>
</tr>
<tr>
<td>Exon17</td>
<td>131</td>
<td>0.909</td>
</tr>
<tr>
<td>Exon18</td>
<td>140</td>
<td>0.757</td>
</tr>
<tr>
<td>Exon19</td>
<td>161</td>
<td>0.738</td>
</tr>
</tbody>
</table>

The exons in red text have low alignment score, which may make the assembling difficult.

Table S2: The description of gene family of peroxisomal biogenesis factor 11 in Arabidopsis

<table>
<thead>
<tr>
<th>Protein ID</th>
<th>Score</th>
<th>Description</th>
<th>Alternative ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT1G01820.1</td>
<td>1</td>
<td>peroxisomal biogenesis factor 11 family protein / PEX11 family protein contains Pfam PF05648: Peroxisomal biogenesis factor 11 (PEX11)</td>
<td>Q9LQ73 (Uniprot)</td>
</tr>
<tr>
<td>AT2G45740.1</td>
<td>0.823</td>
<td>peroxisomal biogenesis factor 11 family protein / PEX11 family protein contains Pfam profile PF05648: Peroxisomal biogenesis factor 11 (PEX11)</td>
<td>O80845 (Uniprot), Q93Z4 (Uniprot)</td>
</tr>
<tr>
<td>AT3G61070.1</td>
<td>0.796</td>
<td>peroxisomal biogenesis factor 11 family protein / PEX11 family protein contains Pfam PF05648: Peroxisomal biogenesis factor 11 (PEX11)</td>
<td>Q84JW1 (Uniprot)</td>
</tr>
</tbody>
</table>
Table S3: The results of 25 core eukaryotic genes and the comparison of 3 whole-genome assemblies

<table>
<thead>
<tr>
<th>Contig</th>
<th>Core Gene</th>
<th>GenomeThreader Report</th>
<th>Contig Length</th>
<th>500bp_K27</th>
<th>500bp_K45</th>
<th>500bp+2kb</th>
<th>AERVX00000000.1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Score</td>
<td>Exon</td>
<td>Coverage</td>
<td>Length</td>
<td>Percentage</td>
<td>Length</td>
</tr>
<tr>
<td>Contig1</td>
<td>7291151_ K</td>
<td>0.84</td>
<td>581</td>
<td>1.00</td>
<td>3,522</td>
<td>2,680</td>
<td>76.1%</td>
</tr>
<tr>
<td>Contig2</td>
<td>7293237_ K</td>
<td>0.62</td>
<td>3,585</td>
<td>0.97</td>
<td>7,472</td>
<td>2,409</td>
<td>32.2%</td>
</tr>
<tr>
<td>Contig3</td>
<td>7292688_ K</td>
<td>0.81</td>
<td>577</td>
<td>0.93</td>
<td>4,366</td>
<td>4,366</td>
<td>100.0%</td>
</tr>
<tr>
<td>Contig4</td>
<td>7293219_ K</td>
<td>0.67</td>
<td>714</td>
<td>0.98</td>
<td>3,027</td>
<td>1,213</td>
<td>39.7%</td>
</tr>
<tr>
<td>Contig5</td>
<td>7293774_ K</td>
<td>0.74</td>
<td>983</td>
<td>0.65</td>
<td>3,232</td>
<td>2,231</td>
<td>69.6%</td>
</tr>
<tr>
<td>Contig6</td>
<td>7294391_ K</td>
<td>0.52</td>
<td>984</td>
<td>0.97</td>
<td>5,972</td>
<td>4,357</td>
<td>73.5%</td>
</tr>
<tr>
<td>Contig7</td>
<td>7294310_ K</td>
<td>0.67</td>
<td>2,526</td>
<td>0.87</td>
<td>13,469</td>
<td>5,976</td>
<td>44.4%</td>
</tr>
<tr>
<td>Contig8</td>
<td>7294336_ K</td>
<td>0.53</td>
<td>767</td>
<td>1.00</td>
<td>7,338</td>
<td>3,832</td>
<td>52.2%</td>
</tr>
<tr>
<td>Contig9</td>
<td>7294421_ K</td>
<td>0.85</td>
<td>1,447</td>
<td>0.70</td>
<td>2,264</td>
<td>2,036</td>
<td>89.9%</td>
</tr>
<tr>
<td>Contig10</td>
<td>7295987_ K</td>
<td>0.54</td>
<td>390</td>
<td>0.65</td>
<td>2,815</td>
<td>2,447</td>
<td>86.9%</td>
</tr>
<tr>
<td>Contig11</td>
<td>7296763_ K</td>
<td>0.57</td>
<td>580</td>
<td>0.76</td>
<td>3,347</td>
<td>1,340</td>
<td>40.0%</td>
</tr>
<tr>
<td>Contig12</td>
<td>7297033_ K</td>
<td>0.51</td>
<td>1,437</td>
<td>0.97</td>
<td>3,244</td>
<td>3,240</td>
<td>99.9%</td>
</tr>
<tr>
<td>Contig13</td>
<td>7297923_ K</td>
<td>0.68</td>
<td>350</td>
<td>0.53</td>
<td>5,201</td>
<td>3,430</td>
<td>65.9%</td>
</tr>
<tr>
<td>Contig14</td>
<td>7298330_ K</td>
<td>0.74</td>
<td>726</td>
<td>0.99</td>
<td>2,440</td>
<td>2,418</td>
<td>99.1%</td>
</tr>
<tr>
<td>Contig15</td>
<td>7294421_ K</td>
<td>0.66</td>
<td>1,800</td>
<td>0.85</td>
<td>5,372</td>
<td>1,455</td>
<td>27.1%</td>
</tr>
<tr>
<td>Contig16</td>
<td>7296330_ K</td>
<td>0.66</td>
<td>597</td>
<td>0.97</td>
<td>2,934</td>
<td>2,916</td>
<td>99.4%</td>
</tr>
<tr>
<td>Contig17</td>
<td>7300729_ K</td>
<td>0.53</td>
<td>427</td>
<td>0.59</td>
<td>4,954</td>
<td>1,361</td>
<td>27.5%</td>
</tr>
<tr>
<td>Contig18</td>
<td>7300832_ K</td>
<td>0.77</td>
<td>552</td>
<td>1.00</td>
<td>1,757</td>
<td>1,073</td>
<td>61.1%</td>
</tr>
<tr>
<td>Contig19</td>
<td>7301071_ K</td>
<td>0.91</td>
<td>384</td>
<td>1.00</td>
<td>6,447</td>
<td>2,145</td>
<td>25.4%</td>
</tr>
<tr>
<td>Contig20</td>
<td>7302800_ K</td>
<td>0.57</td>
<td>832</td>
<td>0.98</td>
<td>5,250</td>
<td>2,723</td>
<td>51.9%</td>
</tr>
<tr>
<td>Contig21</td>
<td>7303207_ K</td>
<td>0.89</td>
<td>1,968</td>
<td>1.00</td>
<td>5,786</td>
<td>4,423</td>
<td>76.4%</td>
</tr>
<tr>
<td>Contig22</td>
<td>7303207_ K</td>
<td>0.79</td>
<td>1,989</td>
<td>0.95</td>
<td>4,768</td>
<td>2,043</td>
<td>42.9%</td>
</tr>
<tr>
<td>Contig23</td>
<td>7303608_ K</td>
<td>0.62</td>
<td>461</td>
<td>0.69</td>
<td>3,838</td>
<td>3,186</td>
<td>83.0%</td>
</tr>
<tr>
<td>Contig24</td>
<td>7303843_ K</td>
<td>0.77</td>
<td>747</td>
<td>1.00</td>
<td>5,111</td>
<td>3,179</td>
<td>62.2%</td>
</tr>
<tr>
<td>Contig25</td>
<td>7304146_ K</td>
<td>0.84</td>
<td>376</td>
<td>0.54</td>
<td>1,427</td>
<td>971</td>
<td>68.5%</td>
</tr>
</tbody>
</table>

### APPENDIX B. SRASSEMBLER INSTALLATION AND USAGE

**Installation**

This program can be compiled as serial or parallel version in Linux distributions. For serial version, simply use:

1. Type 'cd src' to change the current directory to the source folder.
2. Type 'make' to compile the package. The executable file 'SRAssembler' should be generated in 'bin' folder.

3. Type `make install' to install the program to /usr/local/bin.

For parallel version, simply use:

1. Type 'cd src' to change the current directory to the source folder.

2. Type 'make mpi' to compile the package. The executable file 'SRAssembler_MPI' should be generated in 'bin' folder.

3. Type `make install' to install the program to /usr/local/bin.

C++ Boost libraries option

In most linux systems, the C++ Boost libraries are already installed. If yours are not, please download it from. They do not need to be compiled. If these header files are not in /usr/include/boost, please use with-boost option to specify it. Like this:

   make with-boost=boost_path or

   make mpi with-boost=boost_path
Running the SRAssembler

Usage:

SRAssembler [options] -q query_file -s species -P parameter_file [ -f library_file or -l reads_file1 -2 reads_file2 ]

-q: Required. Query file name


-P: Required. Parameter configuration file.

-f: Required if you do not specify -l option. Library file

-l: Required if you do not specify library file. Single-end read file name or the left reads file name for paired-ended reads

-2: Right reads file name for paired-ended reads

-t : Query file type: Options: 'protein', 'cdna' [Default: protein].

-o: Output directory [Default: current directory]

-m: Minimum contig length to be reported [Default: 200]

-M: Maximum contig length to be reported [Default: 10000]

-k: The kmer of assembler. The format is : start_k:interval:end_k. The start_k and end_k must be odd value, and The interval must be even value. For example, '15:10:45' means k-mer value 15, 25, 35, 45 will be tested. [Default: 15:10:45]

-z: Insert size of paired-end reads [Default: 300]

-n: Number of rounds [Default: 10]

-x: Number of reads of split file [Default: 500000]

-a: The round number the assembly starts [Default: 1]
-r: The round number to clean the unmapped reads. For example, "5" means SRAssembler removes reads cannot be mapped to contigs at round 5, 10, 15, 20... [Default: 0 (disabled)]

-v: Verbose output

Some important options:

- **-a**: this option indicates which round we start to do the assembly. For the deeper dataset, we can start the assembly from round 2 even round 1. But for some datasets with lower coverage, assembly in the early rounds may cause the wrong contigs, which will further affect your final results.

- **-n**: the number of rounds. If the length of contigs is not as long as you need, you can increase this value to get longer contigs at the cost of running time.

- **-m**: minimum contig length. For each round, only length of contigs larger than this value will be chosen to be query sequences of the next round. If this value is too high, your true contig may be ignored during recursion.

- **-M**: maximum contig length. For each round, if contig length is larger than this value, the SRAssembler will stop assembling them. All reads associated with this contig will be removed as well, thereby improving the running time.

Library definition file. You can use option `-f` to specify a library definition file.:

Each library is specified by [LIBRARY] section. Each section includes the following items:

- **insert_size**: the insert size of the library. This value is used in paired-end reads.

  Default: 300.
direction : the sequencing direction of paired-end reads. 0 : forward-reverse; 1: reverse-forward. Default: 0.

r1: left-end reads file or single-end reads file.

r2 : right-end reads file. Do not specify if your library is single-end.

format : "fastq" or "fasta". Default: “fastq”.

Note that if your library contains both paired-end and single-end reads, please treat them as two libraries. Here is an example of two libraries:

[LIBRARY]
insert_size=200
direction=0
r1=reads1_200.fq
r2=reads2_200.fq
format=fastq

[LIBRARY]
insert_size=1000
direction=0
r1=reads1_1000.fq
r2=reads2_1000.fq
format=fastq

Parameter configuration file:
You can use option -P to specify a parameter configuration file, which allow users to specify the parameters of the programs used in SRAssembler such as Vmatch, GenomeThreader, GeneSeqer, Exonerate and Snap)

The parameters are grouped by the programs. Like:

[Vmatch_init]
e=1
l=11

[Vmatch]
e=0
l=30

[GenomeThreader]
gcmincoverage=10
prminmatchlen=15

[GeneSeqer]
x=14
y=14
z=25

[Exonerate]
percent=30

[Snap]
snaphmm=A.thaliana.hmm
Note that [Vmatch_init] is the parameter settings for the initial round of SRAssembler.

- **l** option in [Vmatch_init] is the match length in the first round. Since the first round is the alignment of homologous genes, if you set it too high, you may get very few hits. The default value for protein query sequences is 10; 30 for cDNA query sequences. This still depends on the depth of your dataset. If your assembly has very poor spliced alignment results, you can decrease this value to gather more reads to improve your assembly results.

- **e** option in [Vmatch_init] is the mismatches allowed in first round. The default value is 1. If this value is too low, you may get very few matched reads if your query sequences are not very well conserved. On the other hand, when you set this value too high, some false positive reads may be fetched.

- **l** option in [Vmatch] is the match length of recursive round. The default value is 30. This value controls the speed of chromosome walking. If your dataset is very deep, you can specify higher value and save the running time.

- **e** option in [Vmatch] is the mismatches allowed in recursive round. The default value is 0.
ACKNOWLEDGEMENTS

I would like to express my thanks to my advisors, Dr. Brendel and Dr. Honavar for their guidance and support throughout this research and the finish of this thesis. I am also grateful to my committee members for their efforts to this work: Dr. Amy Toth, Dr. Xiaoqiu Huang and Dr. Bing Yang. I would also like to thank my lab mates, Krishnakumar Sridharan, Daniel Standage and Yiming Zhang for their advice. Special thanks to my wife, Wen-yi, for her support and belief in me. Without her, this work would not have been possible.