ENABLING EFFICIENT AND STREAMLINED ACCESS TO LARGE SCALE GENOMIC EXPRESSION AND SPLICING DATA

by

Christopher Wilks

A dissertation submitted to The Johns Hopkins University

in conformity with the requirements for the degree of

Doctor of Philosophy

Baltimore, Maryland

October, 2020

Abstract

As more and larger genomics studies appear, there is a growing need for comprehensive and queryable cross-study summaries. We focus primarily on nearly 20,000 RNA-sequencing studies in human and mouse, consisting of more than 750,000 sequencing runs, and the coverage summaries derived from their alignment to their respective gnomes. In addition to the summarized RNA-seq derived data itself we present tools (Snaptron, Monorail, Megadepth, and recount3) that can be used by downstream researchers both to process their own data into comparable summaries as well as access and query our processed, publicly available data. Additionally we present a related study of errors in the splicing of long read transcriptomic alignments, including comparison to the existing splicing summaries from short reads already described (LongTron).

Primary Reader and Advisor: Ben Langmead

Secondary Reader: Michael C. Schatz

Secondary Reader: Liliana Florea

Acknowledgments

I would like to thank my advisor, Ben, for all the advice, time, and effort put into my training as a PhD student. In addition I'd like to thank both Mike and Liliana for their willingness to serve both on my thesis committee as well as on my GBO committee. Further, I'd like to thank Mike for his advice and input on the LongTron project. Also, I want to acknowledge Leonardo Collado-Torres, Abhi Nellore, Kasper Hansen, Rone Charles, Jonathan Ling, Phani Gaddipati, Geo Pertea, and Leonard Goldstein for their help and/or input on the various projects represented here.

My family and close friends have been a critical support for me during this time and I couldn't have endured without them.

More broadly, I'd like to acknowledge all the various input I've had from multiple professors and fellow students over the years beginning in my undergraduate days and extending until now.

Soli Deo Gloria

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Sequence Read Archive (SRA) is a large and valuable repository of public and controlled-access sequencing data, spanning over 44 petabases and doubling in size every 18-20 months (Langmead and Nellore, 2018) (Figure 1.1). Such archives allow researchers to reproduce past studies, combine data in new ways, and access unique datasets that would otherwise be too expensive or difficult to obtain. But researchers struggle to take full advantage of archived data. There is no convenient way to pose scientific questions against the archives without first downloading and re-analyzing the data, which is very time- and compute-intensive.

The situation is analogous to the early days of the World Wide Web, when content was accessed at well known addresses via transport protocols (FTP, HTTP). The web became vastly easier to use with the advent of search engines: crawlers, indexes, and ranking algorithms made it possible for users to filter the web for content relevant to their queries.

One primary source of this sequence data explosion is the high-throughput short-read sequencing of DNA and RNA molecules. This type of sequencing

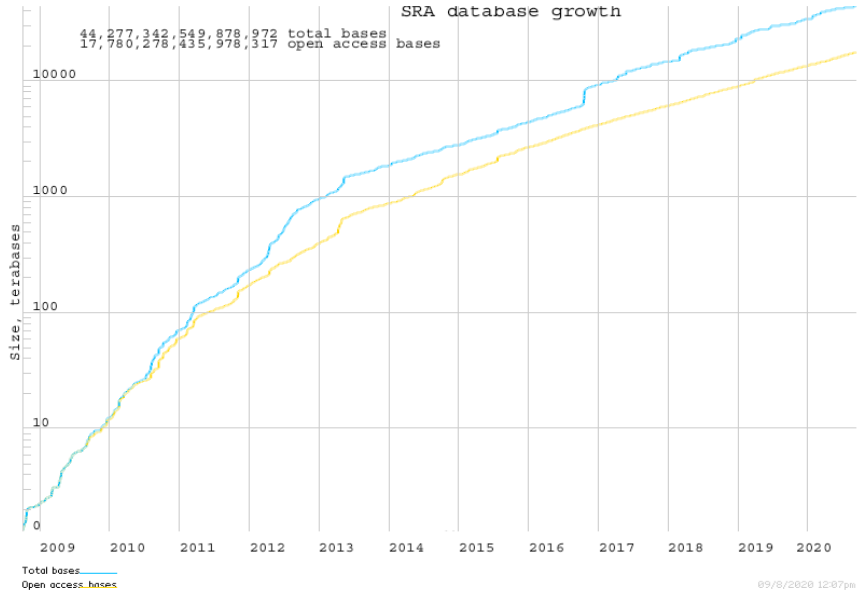

Figure 1.1: Sequence growth in the Sequence Read Archive measured by number of petabases. From https://www.ncbi.nlm.nih.gov/sra/docs/sragrowth/

via fragmented reads, typically 75-250 base pairs long, has been extensively used for research for over 10 years (Dijk et al., 2014). The RNA-seq approach specifically has been utilized for determining gene expression (Bray et al., 2016a; Patro et al., 2017), alternative gene structure (Dobin et al., 2013; Goldstein et al., 2016a), and fusion constructs (Haas et al., 2019), as well as de novo regions of expression throughout genomes of different species (Trapnell et al., 2010; Pertea et al., 2015). Due to continued investment in improving these sequencers over time, the error rates of short reads are relatively low (<1%) while the throughput is high, with up to a terabase coming off a production sequencer in less than 2 days.

In this thesis we describe tools to both efficiently harmonize the alignment and coverage quantification of these RNA-derived short-read sequences, as well as present the data through appropriate layout and indexing techniques

for the downstream biomedical researcher's ease of use and efficient access. As part of this work, we also report on the output of these tools run on more than 750,000 RNA-seq sequencing runs present in the SRA and the Genomics Data Commons (GDC). These sequencing runs are all the human and mouse bulk and smartSeq-related sequencing projects at the time of the start of the Monorail project (October 2019). Further, we address the recent advances in the related field of long read transcriptomics sequencing and its error profiles with specific regard to spliced alignments.

A focus of this work, is the annotation-free alignment of these RNA-seq reads, enabling the discovery of novel transcribed regions. One specific type of novel transcription we are interested in here is novel splice junctions, which use both canonical and non-canonical splice site motifs ("canonical" defined as most represented motifs: GT-AG: 96.5%, GC-AG: 2.6%, AT-AC: 1.0% Nellore et al., 2016a). While the human and mouse annotation represents substantial work and expertise, it is recognized to be incomplete (Zhang et al., 2020; Pertea et al., 2018). Thus for research that looks to find novel biology, a reasonable place to start is to do alignments without referencing any annotation to allow for underlying, potentially unannotated transcription to be found. Figure 1.2 illustrates an alternatively spliced, potentially novel exon.

A second important idea, shared across much this work, is the align once, quantify many times approach we advocate and practice in both recount2 (Collado-Torres et al., 2017b) in the Background section, and in recount3 in the main body of this thesis in the Monorail chapter. This approach requires the generation and persistent storage of per-base coverage BigWig files in lieu of

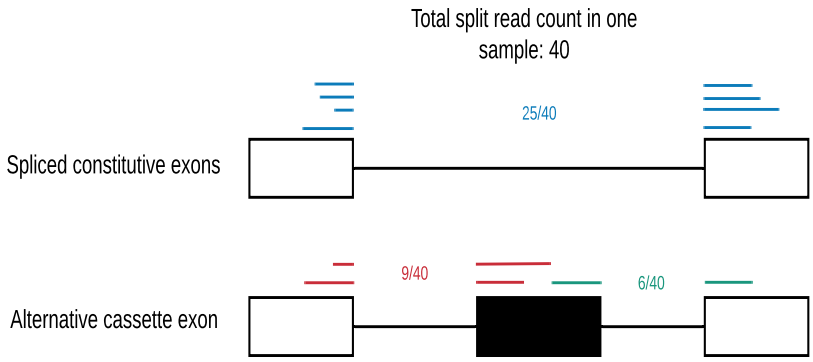

Figure 1.2: Cartoon of a novel, alternatively spliced exon with split read support

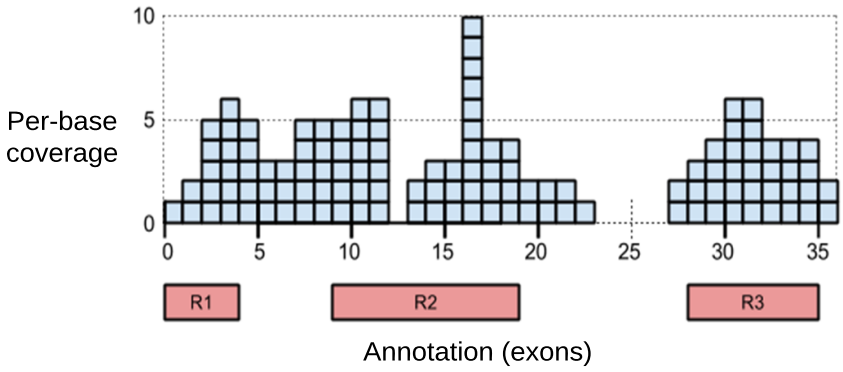

Figure 1.3: Cartoon of the per-base coverage stored in a BigWig with annotation (not stored). Modified from https://github.com/CRG-Barcelona/bwtool/wiki/aggregate

the normal BAM files, which are too large to keep by most groups, outside of the sequence repositories themselves. These per-base BigWig files, combined with our Megadepth tool, allow for the rapid re-quantification over a new or different annotation than what was present when the original alignments were produced. This allows for the far more efficient update of the gene and exon sums, without resorting to re-downloading and re-aligning all the same data again. Figure 1.3 illustrates the data stored in a BigWig file.

1.1 Background

While not the primary focus of this thesis, I worked on multiple other projects foundational to this thesis. Those projects include the Rail spliced-aligner (Nellore et al., 2016a), the dbGaP extension to that project (Nellore et al., 2016c), the analysis of splicing in SRA "Intropolis" project (Nellore et al., 2016b), the recount2 summarising of the coverage data produced by Rail (Collado-Torres et al., 2017b), extensions to that project, recount-brain (Razmara et al., 2019) and finally the FCR2 resource (Imada et al., 2020) which summarized the coverage in recount2 again but over the FANTOM-CAT (v6) set of long non-coding RNAs (lncRNAs). We will describe each of these efforts in slightly more detail in the following subsections.

1.1.1 Rail: a multi-sample aware spliced-aligner

Much of this work builds on both the idea of large scale processing that was pioneered in the Rail project as well as the types of data produced there. Specifically the first version of the Snaptron project indexed and formatted the junction, exon, gene, and ultimately base-level coverage that was generated by the Rail aligner.

Rail had two key concepts which were unique to its alignment approach:

- Borrowing strength across samples
- Leveraging an existing, broadly used parallel architecture MapReduce (Dean and Ghemawat, 2010) for computational efficiency

In the first case, the key innovation was the idea that if alignments were run across multiple samples from the same tissue or cell-type at the same time, the method could "look across" samples for repeated evidence of the same lowly-covered exon-exon splice junctions that were tissue or cell-type specific. This would require that the tissues/cell-types of samples be known beforehand and that Rail would be run on groups of these samples in carefully constructed (not-random) batches of related samples. We showed that this yielded a level of junction-call accuracy comparable to those tools that used an annotation.

The second innovation was a practical one in leveraging the MapReduce architecture of Amazon's Elastic Map Reduce service (EMR) for large scale parallelism in the cloud (AWS). This allowed for the analysis of approximately 100,000 sequence runs, mostly from SRA for human and mouse, as well as GTExV6 and TCGA cancer samples.

The associated project, Rail-dbGaP took the Rail workflow and extended it to support running in a secure fashion in the cloud (specifically AWS EMR). This was a critical component needed to allow for the processing of protected datasets such as TCGA and GTEx. These datasets are directly derived from human cases and are not consented for public release of the raw sequence data. Therefore computing over the sequence required securing the compute environment first. This entailed the use of encryption in both transfer and "at rest" while the data was on the temporary filesystems of the cloud virtual machines. The overall point of this work is still relevant today for the processing of protected data in public cloud environments.

Rail also went a step beyond the typical per-sample aligner in aggregating splicing and other data types (e.g. sequence base indels, not covered here) across samples into summarized matrices. These data were later incorporated into the recount2 and Snaptron projects as well as the splicing analysis in the Intropolis paper. Thus Rail was the source of all the data for these downstream projects.

1.1.2 recount2: bringing large scale transcriptomics coverage data to Bioconductor

The aggregated and per-base coverage data from Rail was an important first step in providing simpler and more usable summaries to the downstream biomedical researcher. However, researchers still needed summaries at the level of gene and exon coverage. They also needed to be able to consume the coverage summaries in smaller slices of the full set of approximately 70,000 sequencing runs from the Rail output. The recount2 project met these needs by summarizing the per-base coverage into exon and gene level counts organized by study and stored in RangedSummarizedExperiment (RSE) objects. Exon-exon junction split read coverage was also organized by study and stored as RSE objects. These RSE objects are widely used in the Bioconductor framework in the R statistical computing platform. Recount2 also provided an interface to access the per-base count BigWig files since this data was too large to be stored as RSE objects.

recount2 has seen wide use by the community evidenced in its count of more than 170 citations in four years and a number of questions asked with the "recount" tag on the Bioconductor support forum. It has also been the

foundation for additional projects, primarily the "recount-brain" and FCR2 projects. In the former project, additional metadata was manually curated for brain specific studies from the SRA and those studies were analyzed alongside brain samples from the GTEx and TCGA projects, both of which had more consistent and complete metadata. In the latter, FCR2 project, the per-base BigWig files from Rail for all of recount2 (approximately 70,000 sequence runs) were re-quantified using the "bwtool" BigWig summing program (Pohl and Beato, 2014) to generate sums across the expanded set of genes (approximately 109,000) in the FANTOM-CAT (v6) annotation, including many lncRNAs. Differential analysis was run across these sums in the various samples to get tumor vs. normal comparisons across the TCGA sums in the FCR2 resource. Additionally, tissue specificity in GTEx was looked at for three categories of lncRNAs compared with a baseline category of mRNA genes.

1.1.3 Intropolis: splicing analysis across 20,000 sequencing runs

Another important output of the Rail project was a vast number of exon-exon splicing calls made by the aligner without reference to any annotation. As our understanding increases about the complexities of the transcriptome and its impact on health, exon-exon splicing emerges as one important correlate of disease (Sveen et al., 2016) (Sibley, Blazquez, and Ule, 2016). Our approach to storing and working with junctions is through a junction-by-sample matrix where the values are raw split-read counts for a junction in a specific sample as illustrated in Table 1.1.

Table 1.1: Example junction-by-sample matrix

Junction	Sample 1	Sample 2	Sample 3
chr1:10-1000	5	0	10
chr3:20-250	1	20	4
chr20:110-300	0	17	8

The term "Intropolis" as a name was determined from the fact that exon-exon splice junctions give rise to introns and the large set of junctions was therefore considered a city of introns. The analysis demonstrated a number of salient points about splice junctions in a large number of samples (approximately 21,000) across multiple disparate studies. A primary result of the Intropolis study was to show that there were still a large number (56,861 or 18.6%) of unannotated junctions which had substantial support (present in $>=$ 1000 run accessions) among sequencing runs in the SRA, many of which were associated with tissue type (Figures 2 and 3 in Nellore et al., 2016b). A junction was considered unannotated if one or both of its splice sites didn't occur together in an annotation. Annotated junctions were derived from a set of nine sources including multiple versions of Gencode across both hg19 and hg38 (lifted over to hg19) listed in Table 1.2.

Another key point of the analysis was showing when new junctions were discovered, correlated with the date of the samples the junctions were discovered in, being added to the SRA repository. Most of the splicing was discovered before 2013, and there was a gradual diminishing of new junctions being discovered by additional samples (Figure 5 in Nellore et al., 2016b).

This analysis continues to influence the work that is presented in the body of this thesis. For both Snaptron and the junction-related analyses we perform,

our set of annotated junctions comes from an expanded version of what's described above. We are also still using the analysis that looks at the total set of unannotated junctions with respect to the number of samples the junction was found in.

1.2 Outline

In the main body of this thesis I will describe three primary projects I worked on while studying for my PhD at Johns Hopkins University. They are as follows

- Snaptron
- Monorail Ecosystem
- LongTron

Table 1.2: Junction annotation sources. Descriptions are from the UCSC Table Browser track detail page or the Gencode website

Short Name	Description	Reference Build
Acembly	AceView gene models constructed from cDNA by Danielle and Jean Thierry-Mieg at NCBI, using their AceView program	hg19
ccdsGene	Human genome high-confidence gene annotations from the Consensus Coding Sequence (CCDS) project	hg19, hg38
Gencode	19 (hg19), 24-26, 29, 33 (hg38)	hg19, hg38
knownGene	A set of UCSC gene predictions based on data from RefSeq, GenBank, CCDS, Rfam, and the tRNA Genes track	hg19, hg38
lincRNAsTranscripts	Human Body Map lincRNAs (large intergenic non coding RNAs) and TUCPs (transcripts of uncertain coding potential)	hg19, hg38
mgcGenes	The Mammalian Gene Collection (MGC) of full-length open reading frames (ORFs) in the genome.	hg19, hg38
refGene	The NCBI RNA reference sequences collection (RefSeq)	hg19, hg38
sibGene	Swiss Institute of Bioinformatics cDNA/EST-based gene predictions	hg19, hg38
vegaGene	Annotated genes from the Vertebrate Genome Annotation (VEGA) database (Human chr14, 20, 22 only)	hg19

The key idea that connects all three of these projects into one cohesive thought is that of aiding the downstream research of exon-exon splicing from RNA-seq short and long read data in human samples. In the first two cases, Snaptron and Monorail, there is another key connection which is efficient computation & summarization over large scale RNA-seq genomics data.

1.2.1 Snaptron

The second chapter extensively covers the custom webservices and query engine, Snaptron, which we built on top of the coverage summaries generated by Rail-RNA. Snaptron facilitates the rapid searching and filtering of millions of exon-exon splice junctions called in 10,000's of samples. This chapter also briefly describes two examples (ASCOT, HUVEC) of multiple collaborations that have involved Snaptron and/or its data in various ways.

1.2.2 Monorail Ecosystem

The third chapter gives a detailed overview of the follow-on projects to Rail, recount2, Intropolis (described briefly earlier) as well as Snaptron itself, together labeled the "Monorail Ecosystem". This work centers around the development and use of the Monorail workflow to further populate and update the recount and Snaptron resources with several hundred thousand more sequencing runs from the SRA, including both human and mouse, bulk and smartSeq single-cell RNA-seq, as well as the latest GTEx (v8) and a re-run of TCGA data.

1.2.3 LongTron

The fourth chapter describes the LongTron project. In the LongTron project we investigated the behavior of errors of spliced alignments of both Nanopore DirectRNA and PacBio IsoSeq transcriptomic long reads aligned with Minimap2 (Li, 2018). This work involved the simulation of transcriptome-derived long reads based on error profiles using the SURVIVOR (StructURal Variant majorIty VOte) tool (Jeffares et al., 2017a). These simulated long reads were then aligned back against the genome and the alignments used to train a random forest model. This model can then be used to help categorize actual long read alignments against the genome into either being "problem-free" or one or more error categories. An additional part of this work was comparing the spliced output of the Minimap2 alignments of both Nanopore and PacBio long reads against short reads in the same or related samples and annotation. This comparison is done at two levels, one with just the splice junctions themselves, and then also at the intron chain level, using a modified version of the gffcompare tool (Pertea and Pertea, 2020). Relatively high concordance was seen at the individual splice junction level, while much less concordance was present at the intron chain level, necessitating the use of a "fuzz" factor (20 base pairs) +/- around the splice sites of the introns. Overall this work should add to the growing body of knowledge of long read behavior in the splicing context.

Chapter 2

Snaptron

2.1 Introduction

Snaptron is a search engine for querying splicing patterns in large, pre-analyzed collections of human RNA sequencing (RNA-seq) samples. Snaptron answers queries via a Representational State Transfer (RESTful) web service interface. Driving Snaptron is a query planner that combines the strengths of different indexing strategies — R-trees, B-trees and term-document inverted indices — to rapidly address user queries.

The data used to service a given query can be a mix of genomic interval data, numeric values associated with genomic intervals, and free- or controlled-text from associated metadata. The REST interface can be queried via HTTP with no software installation necessary. Alternately, Snaptron can be queried via a client script that provides a richer set of queries. Users may also download the (large) files used to populate Snaptron's database as well as the Snaptron server software to create a local Snaptron installation.

While past efforts address the problem of enabling cross-study queries,

most focus on genotype rather than expression or splicing data. GEMINI (Paila et al., 2013) and Genome Query Tools (GQT) (Layer et al., 2016) are complementary tools for indexing and querying genotypes from many individuals, facilitating computation of genomewide summaries over subsets of individuals. BGT (Li, 2016) builds on the positional Burrows-Wheeler Transform (PBWT) (Durbin, 2014) to provide similar functionality, including region-specific queries. The ExAC browser and REST service allow querying of genetic variant frequencies summarized over 90,000 re-analyzed exomes (`http://exac.broadinstitute.org`).

Other past efforts sought to enable querying of expression data in particular. Solomon and Kingsford propose Sequence Bloom Trees (SBTs) (Solomon and Kingsford, 2016) for indexing raw reads from many sequencing samples. They indexed 2,652 human RNA-seq experiments and queried the index using known transcript sequences. The index reports which samples contain the query string, but it is up to the user to build the index, and to reduce a biological question into such presence/absence queries. The Expression Atlas (Petryszak et al., 2016) summarizes a curated subset of ArrayExpress (Kolesnikov et al., 2014) and enables querying of baseline expression and differential expression. But it enables only gene-level queries, and differential expression can only be assessed in certain archived studies.

Another focus of past work has been on computational problems that arise when working with many genomic intervals. BEDTools (Quinlan and Hall, 2010) and GenomicRanges (Lawrence et al., 2013) are widely used tools for working with intervals. The Genome Query Language (GQL) (Kozanitis et al.,

2014) is a SQL-like language for searching and joining aligned reads with other genome-mapped data. GORPipe (Guðbjartsson et al., 2016) provides an interface to genomic interval based data, where queries are accelerated via file seeks on tabular genomic data files sorted by start-positions.

Our aim is to create a full suite of search-engine software for summarizing expression and splicing data. The project began with a "crawling" effort, described previously (Nellore et al., 2016a; Nellore et al., 2016d) that used the Rail-RNA aligner to analyze tens of thousands of RNA-seq samples in a uniform fashion. Snaptron, builds on this by rapidly answering sophisticated queries with respect to splicing, expression data, and metadata. Snaptron makes it easier to leverage large public datasets in day-to-day research. Snaptron is particularly useful for lending additional context and support to hypotheses related to splicing.

In Methods we describe the design of Snaptron and results from the performance profiling that inform the current design. In Results, we describe analyses leveraging both the REST interface and the command-line client for Snaptron. These are examples of analyses that users can perform using Snaptron queries. The Snaptron REST service and documentation are available at: `http://snaptron.cs.jhu.edu`. The Snaptron software is freely available under a Creative Commons Attribution-NonCommercial 4.0 license from: `https://github.com/ChristopherWilks/snaptron`.

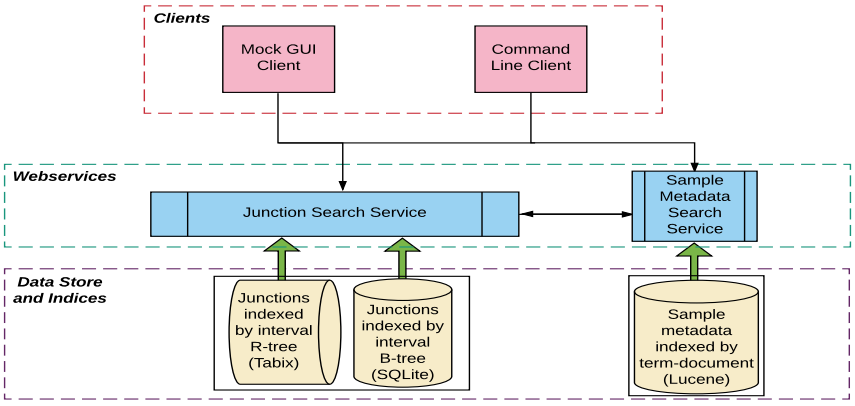

Figure 2.1: The Snaptron architecture consists of three layers (from the bottom up) including data and associated indices (Tabix, SQLite, and Lucene), webservices and processing (Python), and finally the clients (NodeJS and Python). Queries issue from the clients and are processed by web services (black arrows) while responses flow back from the indices through the webservices to the clients (large, green arrows). In addition to junctions, gene, exon and base level coverage is now indexed as well (not shown).

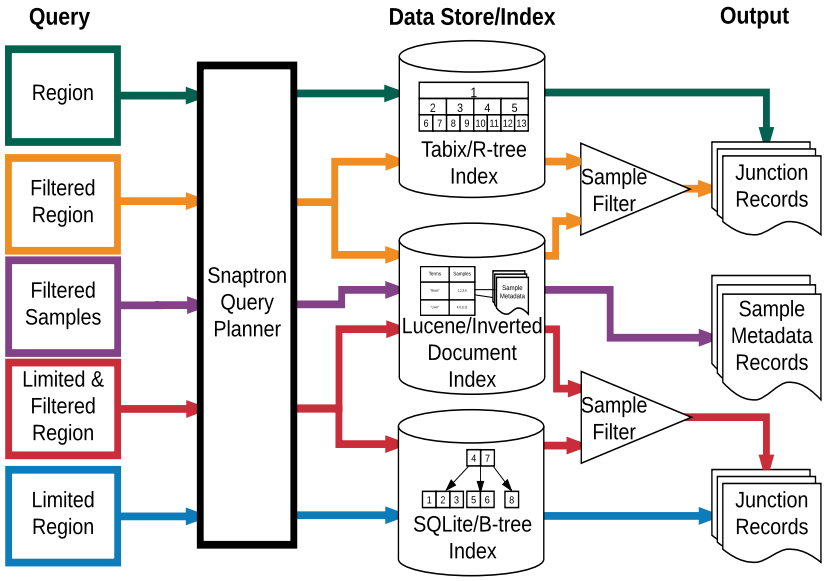

Figure 2.2: The flow of each query through Snaptron and the type of output it produces. Colors correspond to those used in Table 2.1.

Table 2.1: Description of basic and high-level queries supported by Snaptron.

Basic Queries		
Query	**Description**	**Examples in command-line syntax**
Region (R)	Retrieve all junctions lying within a specified genomic interval. A gene name can be given in place of an interval, in which case the interval is taken to be the annotated extents of the named gene. For each returned junction, Snaptron reports a histogram of coverage levels for that junction across all samples with non-zero coverage.	chr1:1-100000 ALK
Region+Metadata (R+M)	Like a Region query but with an additional metadata constraint that limits which samples are considered.	ALK&study_description:cancer
Region+Filter (R+F)	Retrieve all junctions lying within a specified genomic interval but with an additional constraint that might eliminate junctions. The filter can constrain (a) the total, median or average coverage of the junction across samples where it occurs, (b) the number of samples where the junction has occurred, (c) whether or not the junction appears in the Snaptron annotation, (d) the junction's length, i.e. the number of bases spliced out, (e) the junction's strand.	ALK&sample_count>20 ALK&annotated=0 ALK&length>1000&length<2000
Region+Filter+Metadata (R+F+M)	Combining elements of a Region+metadata query and a Region+filter query.	ALK&length>1000& length <2000& tissue:Brain
Metadata (M)	Returns full sample metadata for each sample matching the metadata field query ranked by the Term Frequency - Inverse Document Frequency (TF-IDF) score	library_layout:paired; RIN>8
High Level Queries		
Junction Inclusion Ratio Rank (JIR)	Given two basic junction queries defining two groups of junctions, this returns the list of sample records ranked according to the Junction Inclusion Ratio (JIR) calculated between the two groups. Each returned sample record includes the sample's full set of metadata, total count for the junctions in the first and second groups in that sample, and the JIR for that sample.	See See Appendix C Section 1 for a JIR example script and input
Shared Sample Count (SSC)	Given two groups of junctions, returns the number of samples that have non-zero coverage for at least one junction in both groups. Each group is defined by a basic junction query.	See See Appendix C Section 1 for an SSC example script and input
Tissue Specificity (TS, GTEx only)	Given a group of junctions, returns a tissue specificity table for the set. The group is defined using a basic junction query. The table is N rows by 2 columns, where each row corresponds to one of the 9,662 samples in the GTEx v6 compilation. The first column contains a presence/absence indicator: 1 if every junction in the group is covered in that sample, 0 if not. The second column encodes which of the 32 tissue types the sample comes from.	See Appendix C Section 1 for a TS example script and input

2.2 Methods

2.2.1 Crawling and summarizing

To produce the splicing data served by Snaptron, we used Rail-RNA (Nellore et al., 2016a) to analyze many archived human RNA-seq samples. As has already been described, Rail-RNA is a scalable spliced aligner designed to analyze many samples at once. Among Rail-RNA's outputs is a table summarizing

evidence for exon-exon splice junctions across all samples.

Each row describes a junction, its strand and coordinates, and the number of reads spanning the junction for each sample where it appears. We also created tables detailing metadata for each sample. This is the source material for Snaptron as well as for the `intropolis` resource (Nellore et al., 2016d). `Intropolis` makes junction data available for bulk download but without an indexing facility and without an interface for querying the data. Further details on alignment of GTEx samples are contained in (Nellore et al., 2015), while details on alignment of other SRA samples as well as TCGA samples are contained in (Collado-Torres et al., 2016).

Snaptron further adds auxiliary information to each junction:

- Gene annotation status (discussed below)
- Count of samples with one or more reads covering the junction
- Sum, average, and median of the junction coverage across samples where the junction occurred at least once

Snaptron allows the user to query any of these three compilations of human RNA-seq samples:

- SRAv2: 81M junctions from 44,427 public samples from the SRA
- GTEx: 29M junctions from 9,662 samples from the v6 data freeze
- TCGA: 37M junctions from 11,284 samples from TCGA

SRAv2, GTEx and TCGA use the GRCh38 primary assembly and its coordinates. While raw GTEx and TCGA data are dbGaP-protected, Snaptron's junction-level summaries are, like the SRAv2 compilation, publicly accessible.

We used a composite of several gene annotations (Table 1.2) to determine annotation status of each junction and of each donor and acceptor splice site. If the junction as a whole appears in an annotation, we mark the junction and its splice sites as annotated. If the junction does not appear as a whole, the donor is marked according to whether it is a donor in any annotated junction, and likewise for the acceptor. We used UCSC's `liftOver` tool to convert annotations between genomic coordinate systems.

2.2.2 Data types

Snaptron uses a hybrid indexing approach that enables efficient querying and retrieval. Queries can be concerned with these distinct but related data types:

- *Genomic intervals*, each consisting of a chromosome and beginning and ending offsets. An interval might represent an exon-exon splice junction or an exon as it appears in a gene annotation. A collection of intervals might represent all exon-exon splice junctions in a sample. Intervals within a collection might overlap, i.e. cover some of the same genomic positions.

- *Integer and floating-point numbers*. For example, Snaptron uses non-negative integers to encode the number of reads spanning an exon-exon junction in a sample. Snaptron also stores pre-calculated summaries

such as the average coverage of a junction or the number of samples in which a junction has non-zero coverage.

- *Variable length strings of text*, for sample metadata. Metadata is stored in a combination of semi- and un-structured fields. We call a field semi-structured if it is an accession number or a term from a controlled vocabulary, e.g. sex or tissue type. Unstructured data are free-text fields ranging from phrases to full paragraphs, e.g. a study's abstract or a description of how the sample was prepared for sequencing.

The particular query determines which index or combination of indices Snaptron uses to compose its response.

2.2.3 Region query

A Region (R) query (Table 2.1) retrieves junction data situated in a given genomic interval. It is handled using Tabix (Li, 2011) and its associated R-tree index. Such a query might ask for a list of exon-exon junctions that occur in any sample and that overlap a specified genomic interval. An R-tree index is a tree of nested multi-dimensional bounding rectangles; a node corresponds to the minimum bounding rectangle for points below in the tree. Since we are working with one-dimensional intervals, the bounding rectangles are simply line segments, and the R-tree is essentially an interval tree (Li, 2011) (Kent et al., 2010). A junction and all associated data (including strand, splice motif, annotation status, depth of coverate in each sample) is stored in the lowest R-tree node fully containing the spliced interval.

Querying a human-scale Tabix index on the current Snaptron server takes a few seconds, including the overhead added by Python and Snaptron (Figure 2.3). However, Tabix performance depends on whether the indexed dataset is compressed. We compared the relative performance of the SQLite B-tree and the Tabix R-tree, using Tabix in both compressed and uncompressed modes, and found that uncompressed Tabix outperforms SQLite but SQLite outperforms compressed Tabix (Figure 2.3). Thus, Snaptron uses the Tabix R-tree in uncompressed mode.

When querying junctions that "match" a specified interval, Snaptron allows the user to specify precisely what it means to "match":

- **contains**: match any junction that falls entirely within the specified interval
- **exact**: match any junction with exactly the specified chromosome, start and end coordinates
- **either**: match when either the start or end coordinate falls inside the interval

2.2.4 Filtering attributes

A Region + Filter (R+F) query additionally constrains junction attributes. Attributes describe, for example, annotation status, strand, or prevalence. Examples of attribute constraints are given in Table 2.1. Snaptron uses SQLite and its B-tree index for these queries (`https://www.sqlite.org`). Probes into the B-tree index are efficient — logarithmic in the size of the tree — and the

tree data is organized in a blocked fashion that enables efficient transfers to and from disk. In a Snaptron B-tree, a key represents a junction, including its chromosome, start coordinate and end coordinate. The full record, including the junction and all associated data (strand, annotation status, coverage in each sample, etc) are stored in a related tree structure.

Some junctions in Snaptron's compilations are false positives, due to alignment error and other factors (Nellore et al., 2016a). Consequently, we expect a popular query type will be R+F queries requiring returned junctions to meet a minimum level of prevalence. For instance, a user might request "only junctions with coverage ≥ 50" or "only junctions with non-zero coverage in $\geq$ 1,000 samples."

While Snaptron could re-use the Tabix R-tree for both R and R+F queries, we found SQLite's B-tree was faster for the R+F case (Figure 2.3). This is because the junction attribute filter is not handled by Tabix; instead, the "F" aspect of the constraint has to be handled separately by Snaptron, which parses Tabix output and suppresses records not satisfying the constraint. This adds overhead compared to SQLite, which naturally combines interval (R) and attribute (F) constraints in a single action.

2.2.5 Constraining metadata

Metadata constraints narrow Snaptron's focus to only those samples with metadata matching or containing key phrases. If we think of the junction evidence as forming a matrix with junctions as rows and samples as columns, metadata constraints narrow the query's focus to a subset of the columns.

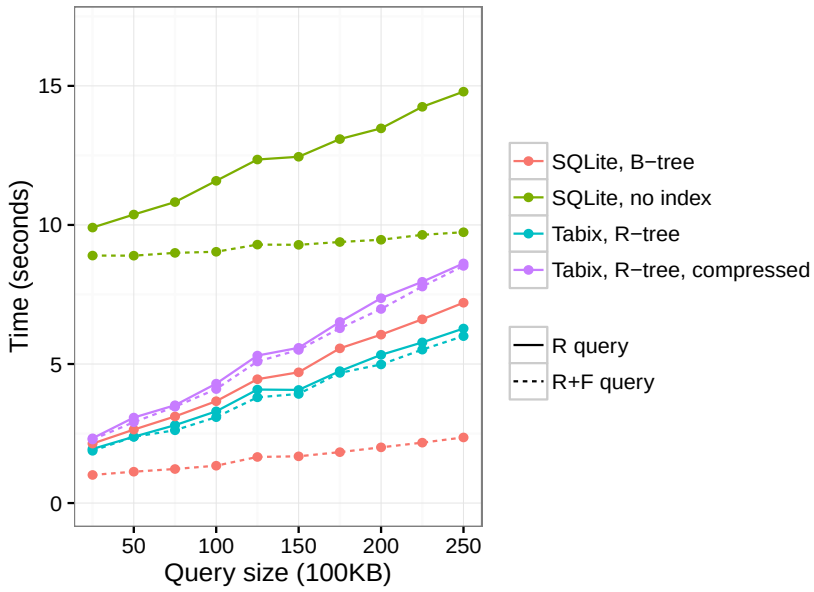

Figure 2.3: Snaptron query wall-clock times for R and R+F queries of increasing size. The queries ask for all (for R) or some (for R+F) junctions overlapping an increasingly large prefix of chromosome 1. The region grows from a 2.5M-base prefix (leftmost) to 25M bases (rightmost) in 2.5M increments. The R+F constraint additionally requires all junctions returned to have `samples_count>=100`. The number of junctions returned by the R query range from 350K for the smallest (leftmost) to 1.5M for the largest (rightmost). The number of junctions returned by the R+F query range from 7.3K for the smallest to 28K for the largest. All data was uncompressed except where noted.

A metadata constraint can be used on its own in a Metadata (M) query, or combined with a Region (R) query or Region + Filter (R+F) query. We call the latter two combinations R+M and R+F+M queries (Table 2.1).

Snaptron uses the popular Lucene (Bialecki, Muri, and Ingersoll, 2012) inverted indexing system to handle metadata constraints. Snaptron includes Lucene indices for each data compilation: SRAv2, GTEx v6 and TCGA. An index associates over fifty metadata fields with each sample. The exact fields depend on the data source. Some fields contain unstructured ("free") text and describe, for example, how the sample was prepared and sequenced or what was being studied. Others are semi-structured, using text labels to describe categorical variables, such as whether the reads are paired-end or the sample's tissue type. For example, the GTEx compilation includes a controlled-vocabulary field describing the tissue of origin, but the SRA compilations do not; that information can often be gleaned from other free text-fields, though sometimes with difficulty (Bernstein, Doan, and Dewey, 2016). The Lucene index allows searching for key phrases in a metadata field.

2.2.6 Query planning

Snaptron's query planner determines the combination of index probes needed to service a query. Region (R), Region + Filtered (R+F), and Metadata (M) queries are each answered from a different index; R queries use the Tabix R-tree, R+F queries the SQLite B-tree, and M queries the Lucene inverted index (Figure 2.1).

The situation is more complex when a query combines region and metadata

constraints, as in R+M and R+F+M queries. Again thinking of a junctions-by-samples evidence matrix, queries combining R and M constraints are concerned with a subset of columns (M constraint) and a subset of rows (R constraint). Such a query might ask for all junctions in the KCNIP4 gene that appear in at least 10 brain samples.

Handling this query decomposes into a few tasks. *Column projection* determines which samples satisfy the metadata constraint. If C denotes the full set of columns (samples), let $C' \subset C$ be the subset satisfying the constraint, determined by querying the Lucene index. *Row projection* determines which junctions (rows) satisfy the region constraint. If R denotes the full set of rows (junctions), let $R' \subset R$ be the satisfying subset, determined by querying the Tabix index. Once C' and R' are known, *submatrix filtration* determines the subset of R' satisfying the "at least 10" constraint. Submatrix filtration is concerned only with the $R' \times C'$ submatrix. Consequently, summaries calculated over the full rows or columns of the original matrix cannot be used here; new summaries must be calculated with respect to the submatrix.

To perform submatrix filtration, sample IDs returned by the Lucene query are converted to an Aho-Corasick automaton. The automaton performs set-wise pattern matching on Snaptron's internal string representation of the matrix rows. Specifically, Snaptron stores a row as a comma-delimited string, with each field containing the concatenation of the sample ID and the read coverage of the junction in that sample. To save space, samples with 0 coverage are not included as fields. The automaton analyzes a single row by consuming the row string's characters one-by-one and signaling when it has

encountered one of the selected columns by entering a special "match" state. Each such match contributes a non-zero entry to the $R' \times C'$ submatrix. Once the submatrix is formed, Snaptron re-calculates row-wise summaries (e.g. sum, average, median). Finally, if an attribute filter (F) was specified, it is evaluated with respect to the recalculated summaries to further narrow the list of returned junctions, completing submatrix filtration.

While Snaptron was originally intended solely for exon-exon splice junction indexing, we eventually extended the interface and indexing to include support for exon, gene, and base-level matrices. In the gene and exon cases, they were formatted according the same set of fields as in the junction case while re-purposing some of the ancillary fields which were splice junction specific to instead contain the gene name and biotype. Both Tabix and SQLite were used to index the genes and exon matrices as in the junction case. However, base-level coverage due to its vastly larger size was only indexed in Tabix.

One reason for the substantially larger size for the base-level matrix, is it's fully materialized, unlike the other three coverage levels which are strictly sparse matrices only tracking samples which had $>= 1$ read coverage counts. The fully materialized matrix for base counts may be somewhat wasteful (approximately 2x more space than the individual BigWigs), but allows for ease of sample sub-selection (projection) for large numbers of samples and bases in a query. A graphic detailing comparing the recount approach to storing base coverage in BigWigs versus the approach used in Snaptron for indexing base-level coverage is in Figure 2.4.

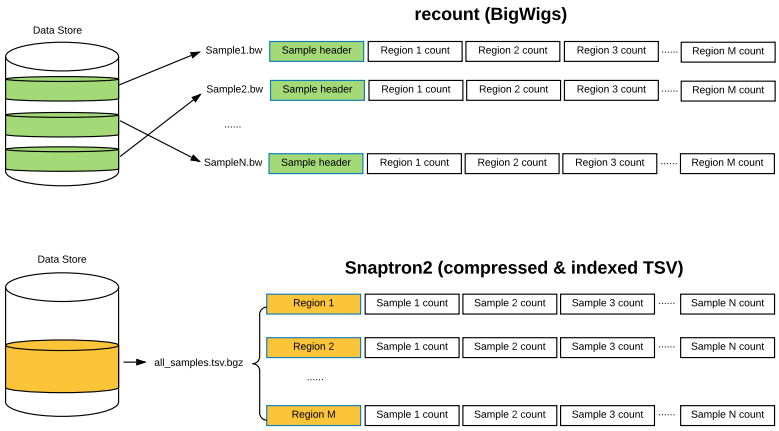

Figure 2.4: Per-base coverage layout in recount (BigWigs) vs. Snaptron (fully pasted and materialized matrices. In theory this highlights the main difference between the two approaches. In practice the base-level coverage is large enough that even the 2nd (Snaptron) approach is stored as slices of the genome in separate files on disk. The overall difference in approach is still maintained.

2.2.7 Higher-level queries

Snaptron supports three queries that we term "higher-level" (Table 2.1) because each involves junction sets defined using sub-queries. The *shared sample count (SSC)* query returns the number of distinct samples with evidence for the co-occurrence of two junctions. As we show later, this is useful for studying prevalence of putative novel exons.

The *tissue specificity (TS)* high-level query uses the GTEx v6 compilation. The user specifies one or more groups of junctions using one or more region sub-queries, one sub-query per group. The TS query returns an $N \times 2$ table, where the N rows correspond to all 9.6K samples from the GTEx project and the two columns correspond to (a) whether a junction from every group occurred in that sample, and (b) which of the 32 GTEx v6 tissue types the

sample was derived from. This list can then be loaded into Python or R to assess tissue specificity.

The junction-inclusion ratio (JIR) high-level query scores each sample according to a particular overrepresented splicing pattern relative to another. The user specifies two groups of junctions using two region sub-queries. Call these groups A and B. The query calculates the normalized difference between coverage counts of the two groups across all samples containing the junctions. This is the “junction inclusion ratio" (JIR) suggested previously by Nellore et al (Nellore et al., 2016d), but with one added to the denominator:

$$\frac{(B - A)}{A + B + 1}$$

A and B represent the total coverage for the two groups of junctions in the sample. Ranking samples according to JIR reveals the degree to which a splicing pattern is specific to a particular kind of sample.

2.2.8 Interfaces

Snaptron provides the following interfaces:

- RESTful web service interface (WSI): handles query requests made by a user or by other Snaptron interfaces via HTTP 1.1. Results come in the form of lists of junctions and associated junction data, or lists of samples and associated sample metadata. Queries usually return within seconds.
- Client command-line interface (CCLI): a Python 2.7 program which handles both basic queries and high-level queries. High-level queries

are decomposed and handled via one or more WSI queries.

- Complete server installation (Local): users can download the underlying Snaptron data and software, build local indices and compilations, and run a local Snaptron service for handling WSI queries. This is for advanced users who require rapid processing of high query volumes. This is also supported by a Docker container image.

- Snapcount: an R interface to the gene, exon, and junction level coverage data in Snaptron as part of Bioconductor. Creates RangedSummarizedExperiments dynamically based on the user's query results.

Users experienced with command-line tools may prefer the direct WSI interface, which has minimal software requirements and responds within seconds in most cases. Users willing to install the lightweight Python CCLI can additionally pose high-level queries. The CCLI can be called from wrapper scripts to compose complex analyses, as shown in our example scripts.

2.3 Results

We describe four applications of Snaptron: three analyses and a simple "mock" graphical user interface (GUI). The analyses leverage public data to provide context or support for a hypothesis about splicing. The simple GUI demonstrates how calls to the Snaptron REST service can be used to facilitate exploratory data analyses and provide images in support of splicing investigations.

Figure 2.5 shows screen captures from the GUI giving results of Snaptron WSI queries relevant to the analyses.

2.3.1 Novel Exon Discovery and Evaluation

Snaptron can measure prevalence of candidate junctions and exons in public RNA-seq data. The candidates need not be annotated; Snaptron's junction calls were made without the influence of a gene annotation, so it can give support for either annotated or unannotated splicing patterns without favoring one or the other. We demonstrate this by following the work of Goldstein et al (Goldstein et al., 2016b), who searched for unannotated cassette exons in Illumina Human Body Map 2.0 RNA-seq data from 16 normal tissues. A cassette exon was called novel if neither edge coincided with an annotated junction, but the entire exon was located within an annotated gene. Goldstein et al discovered 249 novel exons and validated 216 in a separate cohort using additional paired-end RNA-seq sequencing.

To find evidence for these 249 exons, we posed a high-level Snaptron query that (a) gathered evidence for the exons in the SRAv2 and GTEx compilations, and (b) scored the exons according to shared sample count (SSC), the number of samples with evidence for the exon. The query constrained the reported junction's strand to match the strand of the enclosing annotated gene. Further, the query included the "either" modifier to ensure one end of the queried junctions would exactly match the flanking coordinate on either the 5′ or 3′ end. We found that out of 249 putative exons, 236 (94.8%) occurred in both the SRAv2 and GTEx compilations.

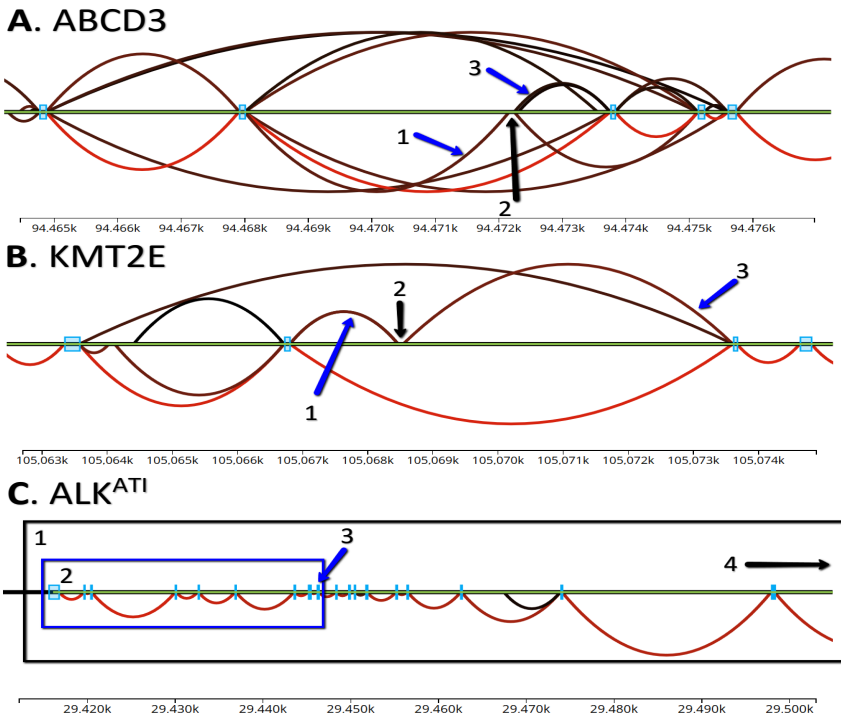

Figure 2.5: Three mock up GUI screen captures corresponding to the three analyses. Green horizontal lines indicate the genome. Arcs indicate exon-exon splice junctions. Colors indicate the number of samples having evidence for the junction, ranging from black (least support) to red (most). Annotated junctions are represented by arcs above the green line, and unannotated junctions by arcs below the line. Light blue rectangles are annotated exons. **A)** Splice junctions matching the Goldstein et al prediction of a novel alternative exon in the ABCD3 gene. A1 is the 5′ junction, A2 is the novel exon, and A3 is the 3′ junction; **B)** KMT2E gene and unannotated junctions supporting a REL exonization event. B1 is the 5′ junction, B2 is the REL exon, and B3 is the 3′ junction; **C)** ALK spliceforms. C1 indicates the full length ALK transcript, C2 is the truncated ALK^{ATI} transcript incorporating only the last 10 exons (ALK is on the reverse strand, and so is laid out right-to-left), C3 is the alternative transcription initiation exon, and C4 is the upstream full transcription initiation site.

Of the 236, 204 (86.4%) were among the exons that Goldstein et al validated in a separate cohort, while the remaining 32 were among the exons that failed to validate. We further used the shared sample count (SSC) to score each of the 236 exons and found that the exons that validated by Goldstein et al had significantly higher SSC than those that failed 2.6. This was true regardless of whether we used the SRAv2 or the GTEx compilation to calculate the score.

This elaborates Goldstein et al's analysis in two key ways. First, Snaptron used public data to score candidate novel exons according to the amount of supporting evidence across tens of thousands of public RNA-seq samples. The scores are valuable both for understanding the degree to which the exons should be considered "novel," and for prioritizing follow-ups such as PCR experiments. Second, Snaptron's comprehensive annotation shed further light on the annotation status of the exons. While Goldstein et al determined that the 249 putative novel exons were unannotated at the time, we found 132 were fully annotated by one or more annotation sources used by Snaptron, with the SIBgenes (*SIBGenes Gene Prediction Track* 2014) and ACEview (Thierry-Mieg and Thierry-Mieg, 2006) tracks annotating most of the 132.

2.3.2 Exonization of Repetitive Elements

Snaptron can use public data to assess tissue specificity of a splicing pattern. A repetitive element locus (REL) exonizaton event is an instance where a stretch of repetitive sequence (e.g. a SINE or LINE) is spliced into a surrounding gene as an exon. A study by Darby et al (Darby et al., 2016) reported numerous REL exonization events in human protein-coding genes, including events specific

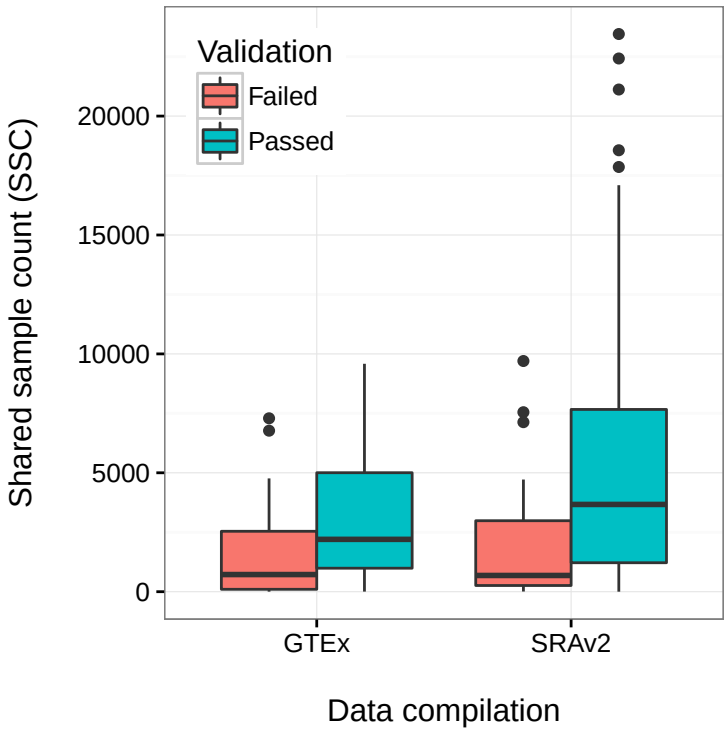

Figure 2.6: Co-occurring sample counts distinguishing validated from non-validating alternatively spliced exons. For GTEx, Wilcoxon rank-sum p = 2e-04. For SRAv2, Wilcoxon rank-sum p = 1e-05.

to brain or blood. We used Snaptron to study these events and to measure tissue specificity with respect to the GTEx compilation.

We first obtained coordinates for 5 PCR-validated REL exonization events in three genes (KCNIP4, KMT2E and GLRB). We then used the shared sample count (SSC) high-level query to measure the prevalence of the events in the SRAv2 and GTEx collections. We noted that the samples studied by Darby et al, derived from the Stanley brain collection, were not present in these compilations. For both the SRAv2 and GTEx compilations, we found that all five events had a shared sample count of 39 or greater. We also found that none of the junctions flanking the events were fully annotated, in agreement with Darby et al. We also analyzed the tissue specificity of the 5 REL exons using Snaptron's high-level Tissue Specificity (TS) query (Table 2.1). We then performed a Kruskal-Wallis rank sum test on the TS query result, using the presence/absence results as the data and the tissue-annotation results as the group labels. All rank sum tests yielded $P < 1 \cdot 10^{-9}$, indicating strong tissue specificity. For example, the REL exon we refer to as `GLRB_1` is present in 33% of the 1,409 samples labeled "Brain" but only 3% of other samples. Similarly, the REL exon `KMT2E_1` is present in 56% of the 102 samples labeled "Bone Marrow" but only 12% of other samples.

2.3.3 ALK and Junction Inclusion Ratio

Snaptron can also be used to study splicing patterns involving many junctions. To demonstrate this, we performed an experiment modeled on Nellore et al's analysis of the anaplastic lymphoma kinase (ALK) gene's ALK^{ATI} variant

isoform (Nellore et al., 2016d). ALK is mutated or aberrantly expressed in some cancers, with its ALK^{ATI} variant, characterized by an alternative transcription initiation (ATI) site, found to be expressed in 11% of melanomas (Wiesner et al., 2015).

Following Nellore et al, we used Snaptron to demonstrate the ALK^{ATI} variant and related EML4-ALK gene fusion can also be found in non-cancer samples. Note that whereas Nellore et al distinguish between the ALK^{ATI} variant and the EML4-ALK fusion by integrating other assays, we do not make the distinction here.

We started by using Snaptron's high-level JIR query to rank samples in order according to the difference between the total coverage of ALK junctions downstream of the ATI versus the junctions upstream. The sets of upstream and downstream junctions are defined using R+F queries. We constrained the strand to be the same as that of the ALK gene and required that junctions lie within ALK's annotated boundaries.

Also following Nellore et al, we postprocessed the JIR results to exclude samples with fewer than 50 total reads covering the ALK junctions. We found that the top 10 samples in our JIR-ranked list exactly match those they reported, including the unexpected melanocyte and macrophage samples.

2.3.4 Client Command-Line Interface

The CCLI is the basis for the three analyses described above. In addition, the CCLI offers two other functions, "psi" and "intersection" which users may find useful in their queries. The first is the ability to provide results for certain

queries in terms of the percent spliced in (PSI) metric common in splicing analyses. The CCLI will list samples ranked by the PSI of a cassette-exon analysis which uses two basic queries, one query defining the included junctions (generating the PSI) and a second query defining the excluded junctions. The second CCLI function supports the formation of general conjunctive queries by grouping basic queries together. The intersection of the resulting junctions from the basic queries is then taken by the CCLI presented as a single list to the user.

2.3.5 Graphical User Interface Application

Finally, we demonstrate that the REST API is powerful enough to enable exploration and visualization of splicing patterns across tens of thousands of samples.

We developed a "mock" GUI with many of the features we expect are needed by typical biological users. Though the GUI is not full-featured and we do not consider it a primary interface for Snaptron, it does allow users to (a) select a gene or region of interest, (b) filter and color-code the junctions according to quantitative summaries like shared sample count and average coverage, and (c) distinguish annotated from unannotated junctions.

Figure 2.5 shows how the GUI presents splicing data relevant to the previous three analyses.

2.4 Discussion

Curated summaries are now available for collections of over 70,000 human RNA-seq samples (Collado-Torres et al., 2016). This motivates the computational question: how do we build systems that make it easy for typical biological researchers to ask and answer questions using these resources? Snaptron is a search engine that combines summarized output from splice-aware RNA-seq alignment tools like Rail-RNA (Nellore et al., 2016a) and Monorail—described elsewhere in this thesis, with a range of indexing strategies and a sophisticated query planner. Snaptron allows researchers to query the vast amount of splicing data now available in summaries like `intropolis` (Nellore et al., 2016d). At no point are users required to download or process raw sequencing data, or any other large files.

Snaptron's design addresses the question of how to combine the best qualities of multiple indexing and database systems in a way that allows rapid queries, even when queries are concerned with a combination of both structured interval and numeric data, and much less structured textual metadata. The design mixes genomics-oriented software like Tabix (Li, 2011) with more generic database and indexing systems like SQLite and Lucene (Bialecki, Muri, and Ingersoll, 2012). Generic systems like SQLite performed surprisingly well, sometimes better than genomics-oriented tools when queries conjoined interval constraints with other constraints.

We used Snaptron to assess: (a) prevalence of putative novel junctions and exons, (b) tissue specificity of novel splicing events, and (c) which public samples exhibit the most divergent splicing patterns for a particular gene.

With the growing popularity of RNA-seq analysis tools that quantify with respect to a given gene annotation (Bray et al., 2016c; Patro, Mount, and Kingsford, 2014), thereby trusting the completeness and accuracy of the annotation, questions about which annotated and unannotated splicing patterns are well supported by public data are increasingly crucial. Snaptron makes it easy to measure support for putative splicing patterns.

In the future it will be important to further optimize Snaptron queries that impose complex constraints on both sample metadata and junction data. While we proposed and implemented an Aho-Corasick-based method that is efficient for some expected queries — with response times measured in seconds or tens of seconds — it is not hard to construct more complex queries that push response times to minutes. The main example of this kind of search type is where thousands of samples are used to further filter a larger R+M or R+F+M query where 100's of thousands of junctions are returned. A specific example would be querying for all junctions on chromosome 1 while also requiring that all junctions appear in samples that have the keyword "tissue" in their description metadata field in the GTEx compilation. The problem is fundamentally difficult since it requires scanning, compiling and summarizing a large fraction of the overall data compilation.

Another future goal is to generalize Snaptron's current support for coverage summaries like mean and SSC to additionally support user-defined functions. There are many possible summaries users could define or that have been proposed in previous studies, "percent spliced in" (PSI) (Venables et al., 2008) being a well known example, which is supported in a limited

way in Snaptron's current set of high-level functions. Allowing user-defined functions would require major changes, but would also obviate the need for Snaptron to individually support a large number of potential summaries.

Finally, it will be important to develop a more full-fledged GUI providing a wider range of functions, such as sample browsing, support for all high-level queries, and the ability to "export" splicing data to other browsers such as the UCSC Genome Browser (Tyner et al., 2016).

2.5 Applications of Snaptron

In the previous sections of this chapter we presented the Snaptron tool and a few examples of how it could be run. In this section we will briefly give overviews of separate pieces of work (both published after Snaptron) that successfully applied Snaptron to biological questions.

2.5.1 ASCOT

Snaptron in this case was applied to the question of alternative splicing leading to tissue and cell-type specific exons, primarily in retinal development in human and mouse (Ling et al., 2020). In the alternative splicing catalog of the transcriptome (ASCOT) project we leveraged and expanded the set of splicing data across Snaptron including hundreds of sequencing runs from the ENCODE shRNA-seq knockdown sequencing project (Sloan et al., 2016; Sundararaman et al., 2016) (1159 run accessions) as well as a set of purified mouse tissue and cell-types sequenced via bulk RNA-seq which we labeled "MESA" (732 run accessions from SRA). We further extended the features of

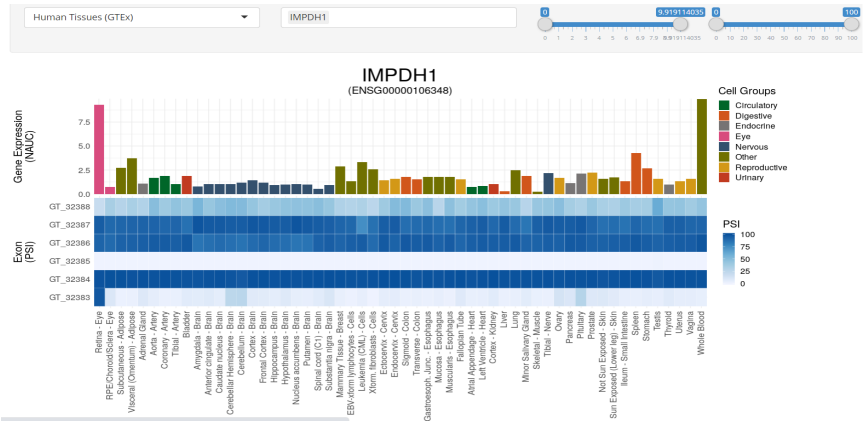

Figure 2.7: IMPDH1 example gene containing a novel exon found in the ASCOT analysis and shown in the ASCOT interface

the Snaptron query engine with the following:

- Merge functionality allowing multiple different Snaptron compilations (sharing the same reference, e.g. HG38) to be queried at the same time with resulting junctions coalesced between the results so all junctions returned would be unique but have the union of samples across compilations they appeared in
- Batch processing of queries supporting a more efficient approach to running thousands of queries against a remote Snaptron server

The ASCOT project then built upon the outputs of Snaptron queries to produce lists of cassette exons (including putative novel ones) based on binary splice events (two spliced in junctions vs. a single skipping junction). Further, these novel exons then were scored using the Percent Spliced In (PSI) metric for comparison across samples, resulting in the further summarized data backing the ASCOT graphical user interface (Figure 2.7).

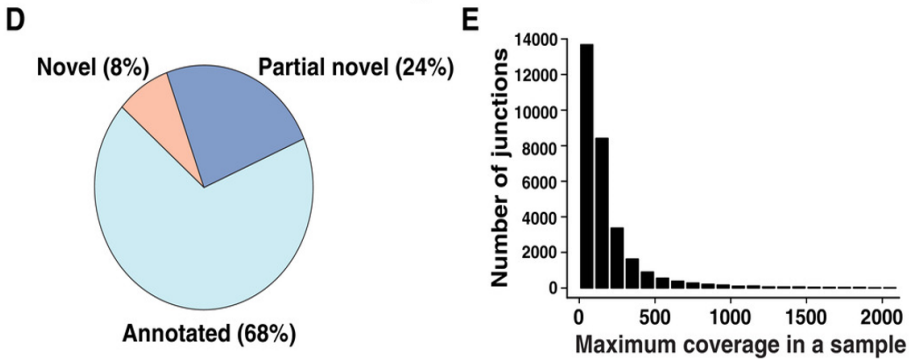

Figure 2.8: Figures 2D and 2E from (Madugundu et al., 2019) showing the breakdown of annotated vs. novel percent of junctions and their split-read counts in Snaptron.

2.5.2 Confirmation of novel splice junctions found in HUVEC tissues alongside proteomics

Snaptron was also used to confirm the existence of a number of alternatively spliced junctions in human umbilical vein endothelial cells (HUVEC) from bulk RNA sequencing (Madugundu et al., 2019). This study performed an integrated analysis of these cells across both RNA-seq and mass spectrometry of proteins generating both splicing data at the mRNA transcript level as well as associated proteomics data. Here all 3 compilations (SRAv2, GTEx, and TCGA) in Snaptron were queried. Figure 2.8 taken from Figures 2D and 2E from the paper show the percent breakdown of novel, partially novel, and fully annotated junctions and their split-read counts found via Snaptron.

Chapter 3

Monorail Ecosystem

3.1 Introduction

We now turn to the full rebuild and regeneration of the recount3 and Snaptron2 resources via the entirely new Monorail workflow. The amount of human RNA sequencing runs available in the sequence read archive (SRA) had more than tripled in the 4 years since Rail was run. This was expected given the high rate of growth the SRA undergoes (Langmead and Nellore, 2018). The recount2 and Snaptron tools needed to be updated with the new human RNA-seq data as well as expanded to include mouse RNA-seq information.

The new version of recount includes a total of 315,449 human and 416,903 mouse samples collected from the SRA, GTEx v8 release (19,214 samples from 972 individuals and 32 tissue types), The Cancer Genome Atlas (TCGA) (11,348 samples from 10,396 individuals and 33 cancer types) and the Cancer Cell Line Encyclopedia (CCLE) (1009 sequence runs). This is substantially more than the 70,603 human RNA-seq samples included in our previous recount2 resource. By extension, recount3 also includes improved queryability. By adding the

snapcount Bioconductor (Huber et al., 2015) package which integrates an updated Snaptron (Wilks et al., 2018), users can perform rapid queries across all summaries at once, e.g. across all the 315K human SRA samples. Such queries enable projects with a specific regional focus — i.e. that study specific genes or splicing patterns — or that are concerned with the prevalence or specificity of an expression or splicing pattern over many studies (Ling et al., 2020; Madugundu et al., 2019; Burke et al., 2020). Users can do this from the command line or from the Python or R programming langauges.

To demonstrate recount3, in Section 3.3.2 we survey splicing patterns across the resource and study the fraction of exon-exon splice junctions that are present in widely used gene annotations for human and mouse.

Also in Section 3.3.2 we show the degree to which recount3 captures cell-type specific splicing across several mouse cell types and find that cell-type-specific junctions are less likely to be present in gene annotations than junctions overall. Finally, in Section 3.3.3 we demonstrate how our base-level coverage summaries reveal examples of non-coding and unannotated tissue- and cell-type specific transcription.

In Section 3.5 we describe the new, Snakemake-based (Köster and Rahmann, 2018) analysis workflow, Monorail, used to produce the summaries which is now much easier to use. It runs from a single Docker/Singularity image and there is a prototype version under development for the popular Galaxy (Afgan et al., 2018) system. We also made several improvements to the design, enabling more regular updates of the resource.

3.2 Background and Related Work

For recount2, we previously analyzed 70,603 sequencing runs from the Sequence Read Archive (SRA), GTEx project (The GTEx Consortium, 2013), and TCGA consortium (Network et al., 2013), compiling splice-junction, gene, exon, and per-base coverages into the recount2 (Collado-Torres et al., 2017b) and Snaptron (Wilks et al., 2018) resources. Other projects have worked to summarize public RNA-seq datasets, with most providing only gene- and transcript-level summaries. ARCHS4 (Lachmann et al., 2018) used the Elysium web service (Lachmann, Xie, and Ma'ayan, 2018) – which in turn used Kallisto (Bray et al., 2016b) – to quantify isoforms in 187,946 human and mouse run accessions from GEO and SRA. ARCHS4 was later updated to include over 520K accessions. The DEE2 project used STAR (Dobin and Gingeras, 2016) and Kallisto to produce gene- and transcript-level summaries for 580K run accessions, later growing to over 1 million, spanning human, mouse and seven other model organisms. Tatlow et al. (Tatlow and Piccolo, 2016) used Kallisto to analyze approximately 12K TCGA and CCLE (Barretina et al., 2012) samples, also producing gene- and transcript-level summaries.

Toil (Vivian et al., 2017) used STAR, Kallisto and RSEM (Li and Dewey, 2011) to generate both spliced alignments (BAM files) and information about splice junctions detected (BedGraphs files). However, it was only run on approximately 20K samples, including TCGA, TARGET, and a previous version of GTEx (about 7K samples).

Other projects have, like recount, produced larger and more multi-purpose summaries from archived RNA-seq datasets. RNAseq-er (Petryszak et al.,

2017) uses the iRAP pipeline to continually analyze new RNA-seq datasets deposited in the European Nucleotide Archive. The effort has produced CRAM, BigWig and bedGraph summaries for over 1 million run accessions to date, which are accessible via a REST API. The Expression Atlas (Papatheodorou et al., 2020) draws on datasets from Geo (Barrett et al., 2013) and Array Express (Athar et al., 2019) to form a compilation of over 1M RNA assays – mostly microarray-based but also many RNA-seq – from multiple species. RNA-seq accessions are analyzed with iRAP. The Single Cell Expression Atlas (Papatheodorou et al., 2020) extends the facility to include over 100 single-cell RNA-seq studies from several species, using Alevin (Srivastava et al., 2019) for analysis.

3.3 Results

3.3.1 Improvements to the resource

We developed a new distributed analysis system called Monorail (see Methods). Using Monorail, we analyzed and summarized over 763K human and mouse sequencing runs, including GTEx V8, TCGA, and 732,352 runs from the Sequence Read Archive (Table 3.1), 416,903 of those from mouse. Altogether, recount3 contains 10 times more run accessions than recount2. In compiling recount3, we processed almost 1 PB of compressed sequencing reads, used approximately 25K node-hours of computation and produced over 150 TB of summarized data (Tables 3.1 & 3.2).

We expanded the types of summaries provided compared to recount2. Previously, we produced gene- and exon-level quantifications with respect to

the Gencode v25 annotation, a BigWig file encoding base-level coverage, and a file describing all of the exon-exon splice junctions detected by the spliced aligner and the number of spanning reads for each.

In recount3, we used STAR (Dobin and Gingeras, 2016) to detect and report exon-exon splice junctions where the donor and acceptor motifs are similar but not identical to a canonical motif. Further, we expanded the gene annotations used to produce gene- and exon-level quantifications; we now quantify each human run using each of four annotations: Gencode v26, Gencode v29, FANTOM-CAT v6 and RefSeq v109, expanding users' ability to study both coding and non-coding RNAs in human. For mouse, we quantify each run with Gencode M23. recount3 also now includes approximately 311,000 (97,000 human; 214,000 mouse) single-cell sequencing runs that used whole-transcript protocols such as Smart-seq (Goetz and Trimarchi, 2012) and Smart-seq2 (Picelli et al., 2013).

We also integrated the Snaptron (Wilks et al., 2018) system for indexing and querying recount3 summaries. Further, we added a new R/Bioconductor interface called `snapcount`, which uses Snaptron to query recount3 summaries. With the addition of the `snapcount` package, it is now easier for users to discover relevant datasets based on metadata, to download summary data at the study or run level, and to obtain results within or across studies in metadata-rich `SummarizedExperiment` objects.

The Monorail system is available to users both as an open source suite of software, and as a self-contained public Docker image that produces identical results.

Table 3.1: Monorail Runs (*includes BAMs for brain tissues **unique jxs)

Compilation	Input Size (TB)	Output Size (TB)	# Sequence Runs	# Studies	# Junctions	# BigWigs (M)	Processing Wall Time (h)
SRA Human v3	474	72	316,443	8,677	228	1.2	1,728
SRA Mouse v1	362	62	416,803	10,088	148	1.7	1,608
TCGA	75	7	11,348	1	31.5	0.045	170
GTEx V6	35	6.7*	9,911	1	22	0.040	168
GTEx V7 & V8	46	4.9	9,303	1	10.6 (new)	0.037 M	123
Total	**992**	**152.6**	**762,939**	**18,768**	**440.1 (396**) M**	**3.022 M**	**3,797**

3.3.2 Human and mouse splicing in SRA

Using recount3 splice-junction summaries, we surveyed unannotated splicing in the SRA. We previously measured this in human using about 20,000 run accessions (Nellore et al., 2016b), but the expanded recount3 resource allows us to use an order of magnitude more run accessions and to study both human and mouse. Further, we now use an updated and expanded set of gene annotations, including multiple versions of Gencode (e.g. V33) and CHESS 2.2 (Pertea et al., 2018). We considered the subset of junctions that appear in at least 5% of SRA run accessions (15,773 out of 315,449 samples for human or 20,846 out of 416,903 for mouse). We found that about 16% of human junctions and 12.5% of mouse junctions were not present in any tested annotation (Figure 3.1). Of the junctions in this subset, about 5% (human) and 3.5% (mouse) had both donor and acceptor sites present in the annotation, but not associated with each other, indicating an exon skipping or similar event. About 8.5% (human) and 7% (mouse) had either the donor or the acceptor present in the annotation, but not both. Remaining junctions (2.5% for human, 2% for mouse) had neither donor nor acceptor annotated. The 5% threshold is chosen to obtain junctions that might be considered "common"; we tested

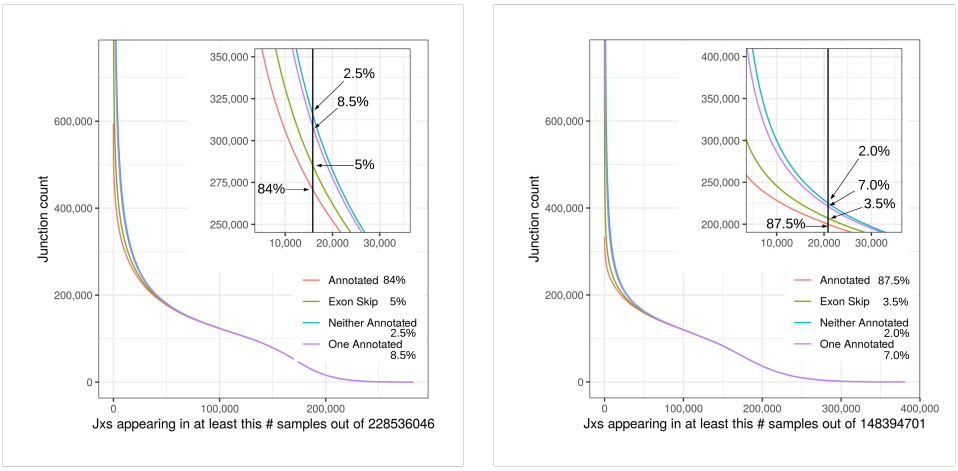

Figure 3.1: Intropolis junction fraction-annotated plots for 1) Human (left) 2) Mouse (right).

other thresholds in Tables A.3 & A.4 in Appendix A.

We next asked whether cell-type-specific splicing patterns tend to be annotated or unannotated. In the ASCOT study (Ling et al., 2020), we asked a similar question while focusing on cassette exons and on datasets where cell type was purified using fluorescence-activated cell sorting (FACS) or affinity purification.

With recount3, we adapted this analysis to consider all splice junctions (not only cassette exons) and by additionally asking: what fraction of cell-type-specific splice junctions are present in any annotation? We considered the same purified datasets as the previous study, which included neuronal cell types, pancreas, muscle stem cells, CD4+ T-cells, B-cells, as well as ovary, testes, kidney, and stomach tissues among others.

For each junction that occurred in at least one sample, we tested its cell type specificity using a Mann-Whitney U test comparing coverage within a

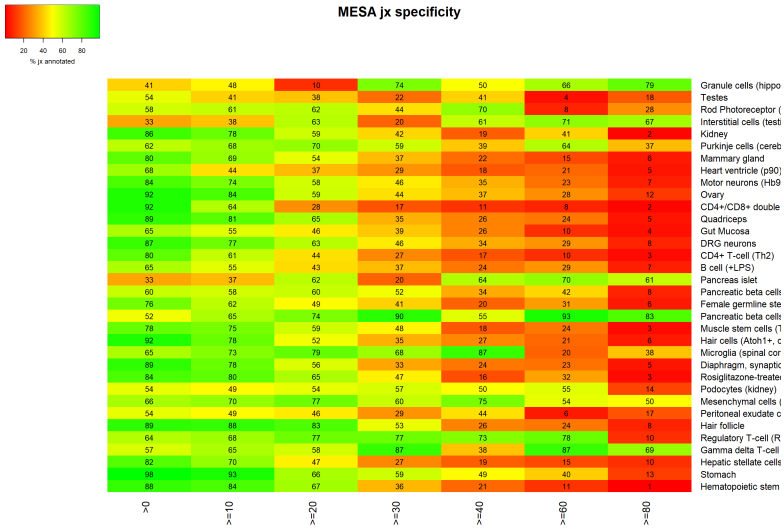

Figure 3.2: MESA cell-type specific enrichment of novel junctions

cell type to coverage in all other cell types (403 samples in 34 studies). We binned the resulting -10 log p-values and calculated the percent of junctions in each bin that appeared in any tested gene annotation (including GENCODE versions M1 – M23 and others). We observed that more cell-type-specific junctions (toward the right) are less likely to appear in annotation (redder color) (Figure 3.2). This suggests that the more specific a splicing pattern is to a particular cell type, the more likely it is to be ignored by the annotation-quantification analyses used to compile other resources such as ARCHS4 and DEE2.

3.3.3 Non-coding and unannotated transcription

Since recount3's BigWig files can be inputs to software for compiling gene and exon-level quantifications, we can quantify recount3 with respect to a new gene annotation without re-aligning the reads. This facilitated our

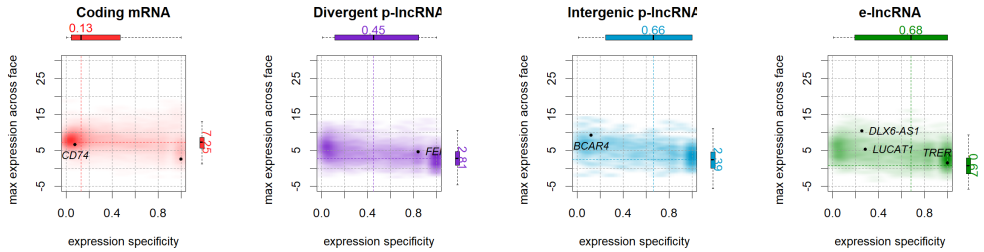

Figure 3.3: Smooth scatter plot showing tissue specificity and overall expression level of different classes of human coding and non-coding mRNAs from the FANTOM-CAT annotation. Measurements are using the GTEx8 compilation. Consistent with past work, non-coding RNAs exhibit a more tissue-specific pattern of expression, indicated by the points' rightward shift relative to the coding mRNAs.

generating the four quantifications included in recount3, range from smaller, more stringent annotations (RefSeq, O'Leary et al., 2016), to more inclusive annotations (GENCODE, Frankish et al., 2019), and to annotations focusing on 5' boundaries and non-coding RNAs (FANTOM-CAT, Hon et al., 2017).

The advantage of diverse annotations is illustrated by the FC-R2 study (Imada et al., 2020), which quantified recount2's bigWigs using the FANTOM-CAT annotation, which includes a large number of non-coding RNAs (Hon et al., 2017). The study reported the tissue specificity of different classes of RNA: coding mRNA, divergent promoter lncRNA, intergenic promoter lncRNA, and enhancer lncRNA. Using recount3's FANTOM-CAT quantifications, we updated that analysis to use the recount3 quantifications, including the additional runs present in GTEx V8 (FC-R2 used about half as many runs). These results confirm those of the earlier study: while ncRNA expression is lower than that of protein-coding genes, ncRNAs tend to have more tissue-specific expression patterns.

To further show the utility of coverage-level summaries, we consider a

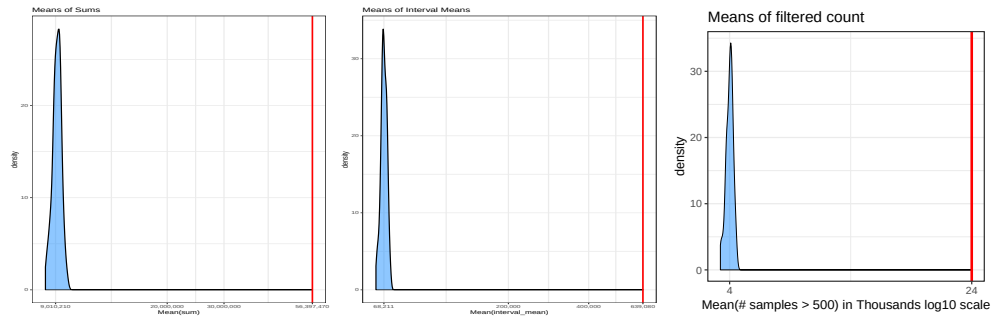

Figure 3.4: SRAv3 with the intervals corresponding to the reannotated ER intervals from (Zhang et al., 2020) and 99 sets of length & chromosome matched random ERs.

recount2-based study by Zhang et al., 2020. With the premise that cell-type specific splicing patterns are less likely to be annotated, the authors used `derfinder` (Collado-Torres et al., 2017a) to analyze 41 GTExV6 tissues and identify genomic intervals that were not present in any gene annotation but that were transcribed in a tissue-specific way. They found several such regions and used other sources of evidence (conservation, genetic constraint, protein coding ability) to argue that the discoveries are not artifacts. Here we further use the BigWig files for the SRAv3 compilation to show that the intronic ERs identified by Zhang et al have substantially more coverage in SRAv3 compared to length-matched, randomly chosen intronic intervals (Figure 3.4). The availability of coverage summaries thus provides a unique facility for studying and validating transcribed regions of the genome that are not necessarily annotated and that don't necessarily involve splicing.

3.4 Discussion

recount3 is a large, easy-to-use resource for querying and obtaining summaries of public RNA-seq datasets. It improves on recount2 in several ways; it

includes an order of magnitude more run accessions, including all of GTEx V8, and approximately 311,000 single cells. It extends the resource to include both mouse and human accessions. It provides powerful interfaces through the Snapcount R package and the updated Snaptron web service. It includes more data types, including comprehensive QC data, and gene- and exon-level quantifications based on various annotations, including FANTOM-CAT. Finally, it uses the new Monorail system for analysis, which is comparatively easy for users to run, and which betters facilitates continual runs on new datasets.

The analyses used to produce recount3's spliced read alignments do not use a gene annotation, and so lacks any bias against unannotated splicing patterns. This makes recount3 an especially appropriate resource for studies that cannot assume that the relevant splicing patterns are annotated (Nellore et al., 2016b; Zhang et al., 2020; Ling et al., 2020). Building on the work of a prior study (Ling et al., 2020), we found that highly cell-type-specific splicing patterns are less likely to be annotated.

While we demonstrated the utility of the recount3 BigWig summaries, they are not yet indexed, and so are not queryable in the same way that gene-, exon- and junction-level summaries are queryable. In the future, it will be important to find space-economical and efficient ways to index and query hundreds of thousands of BigWig files.

Though recount3 includes hundreds of thousands of run accessions from the SRA, the utility of these datasets is often hampered by unreliable or missing metadata. This points to multiple directions for future work. First, it

will be important to continue to build better models for predicting missing metadata and correcting mistakes in metadata (Ellis et al., 2018a). Second, it will be important to enable users with more detailed knowledge of the datasets to create their own collections of related datasets, possibly with their own hand-curated metadata. Finally, since metadata can sometimes be an unreliable way to find relevant datasets, we also think it will be important to design methods that search for related datasets based on their contents rather than their metadata, e.g. using genomic sketching (Baker and Langmead, 2019; Ondov et al., 2016).

3.5 Methods

3.5.1 Design

3.5.1.1 Grid design

Monorail's design follows the grid computing model. In this model, a large-scale computational task is centrally scheduled and orchestrated, with units of work being distributed to computers that might be spread across the world. In our case, orchestration is handled by a collection of services that run continuously in the Amazon Web Services commercial cloud. The computing work was conducted on a few different high-performance computing clusters: the Stampede 2 cluster, is located at the Texas Advanced Computing Center (TACC) and was accessed via the National Science Foundation's XSEDE network. One of the clusters consisted of compute instances rented from the AWS cloud's Elastic Compute Cloud service. And the third cluster was the Maryland Advanced Compute Center located at Johns Hopkins University.

3.5.1.2 Quality control and Alignment

The Monorail analysis pipeline uses various standard tools for analyzing RNA-seq data and compiling QC measures. In particular, Monorail uses the STAR spliced aligner (Dobin and Gingeras, 2016) to align RNA-seq reads in a spliced fashion to the reference genome.

Beside producing alignments, STAR also outputs copious summary statistics that can be used as QC measures. For example,

We use seqtk (Li, 2020 (accessed August 18, 2020)) to compile QC statistics relating to the base composition and base qualities

3.5.1.3 Transcript quantifications

In addition to our traditional alignment approach using STAR we also produce a form of the popular pseudoalignment-based quantification via Salmon. This is included primarily to provide an easy way to compare with other workflows which only use pseudoalignment tools to produce their results for those who want it. In a similar vein, we provide gene and transcripts counts from featureCounts to support comparison with read-based counting workflows.

Monorail consists of three components:

(1) orchestration (Figure 3.5, left), (2) analysis (middle) and (3) aggregation (right).

The orchestration component contains a database describing work to be done along with pointers to inputs and indexes. It coordinates the work via centralized services running in the AWS cloud, including a work queue and

log aggregator.

The analysis component is compute-intensive, involving sequence extraction/decryption, alignment, and quantification as well as other tasks. Our grid-based design allows this step to run in many computing environments at once, exploiting parallelism within and across clusters. Finally, the aggregation component gathers summaries output by the individual analyses and creates per-study (and higher level) tabular summaries for gene-, exon-, and exon-exon junction-level expression.

When starting a new analysis project, each input dataset (e.g. run accession) is combined with information about which analysis workflow and reference data to use, yielding a job descriptor. Descriptors are loaded into the central database as well as a centralized job queue. Analysis nodes are recruited and directed to enter a "job loop" wherein they repeatedly query the queue for the next descriptor (Figure 3.5). Upon dequeuing a descriptor, a parallel Snakemake workflow executes the analysis (Figure 3.6). An analysis node processes jobs until the queue is exhausted or until the local lease expires; e.g. many clusters impose a 2- or 4-day time limit on jobs. Outputs from successful jobs are stored temporarily in cluster-specific scratch storage, then transferred in batches via Globus to the aggregation cluster. Individual Snakemake processes can use multiple threads so there is a degree of parallelism in the three levels of overall data flow—compute node, container process, and, program thread (Figure 3.7).

The output of an analysis of an input dataset consists of gene-, exon-, and junction-level summaries, QC statistics, as well as a BigWig coverage

track and other outputs. The aggregation component takes the gene-, exon- and junction-level summaries from individual datasets and combines them into study-level Tables 3.8. Gene and exon summaries are based on set of human and mouse annotations (e.g. Gencode, listed in Appendix A). Final output consists of gene, exon, and exon-exon splice junction coverage sums, summarized over all samples in the project. Per-base coverage sums are available as BigWig files per sample, due to size. These BigWig files enable the rapid re-quantification of gene and exon sums against new annotations and/or additional unannotated regions of the genome without necessitating the full re-alignment of any sample. Summarized datasets are hosted on SciServer for direct-file based access and separately indexed for region-level querying in Snaptron.

Each of Monorail's components send logging messages to the orchestration component, where they are coalescesed into an AWS CloudWatch dashboard. The dashboard (Figure 3.9) helps to identify performance issues, reducing time spent on debugging and on running jobs when prevailing conditions (e.g. contention for the internet uplink at the archival data source) are not favorable.

3.5.2 Monorail Performance

We used Stampede2, AWS EC2, and an institutional cluster (MARCC) to process approximately 760,000 human and mouse sequencing runs comprising nearly 1 PB of data over six months starting October 2019 (Table 3.2). We used about 25,000 node hours in total, or 0.066 node-hours per sequencing run. We

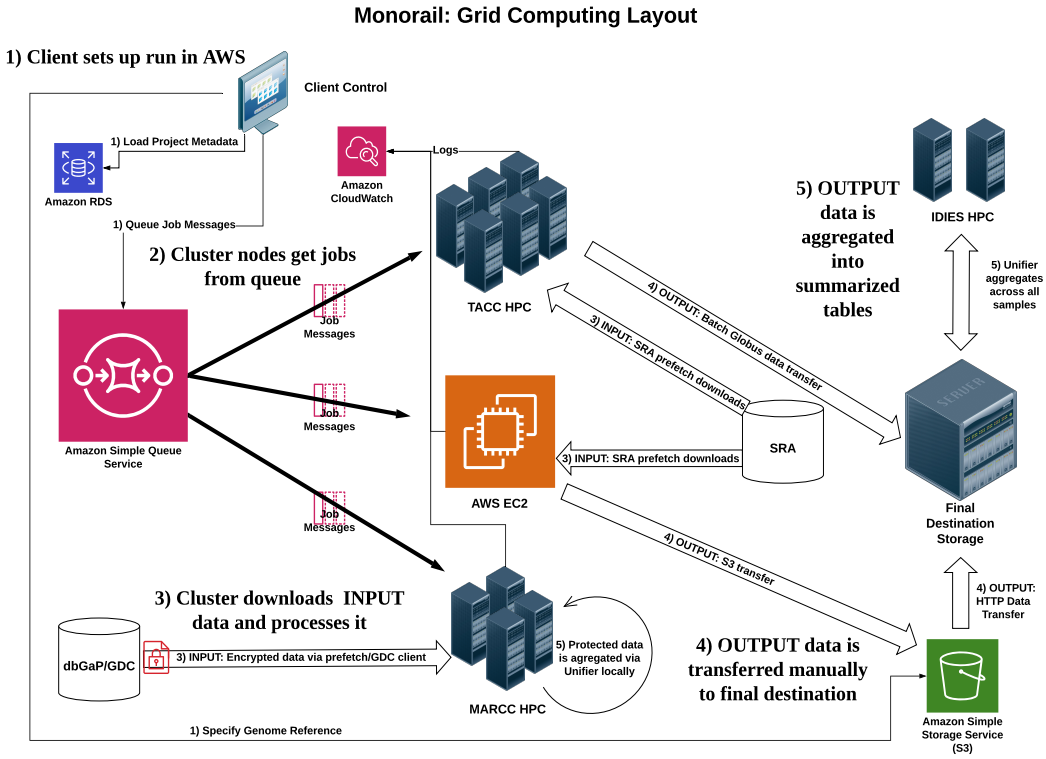

Figure 3.5: Monorail as Grid Computing

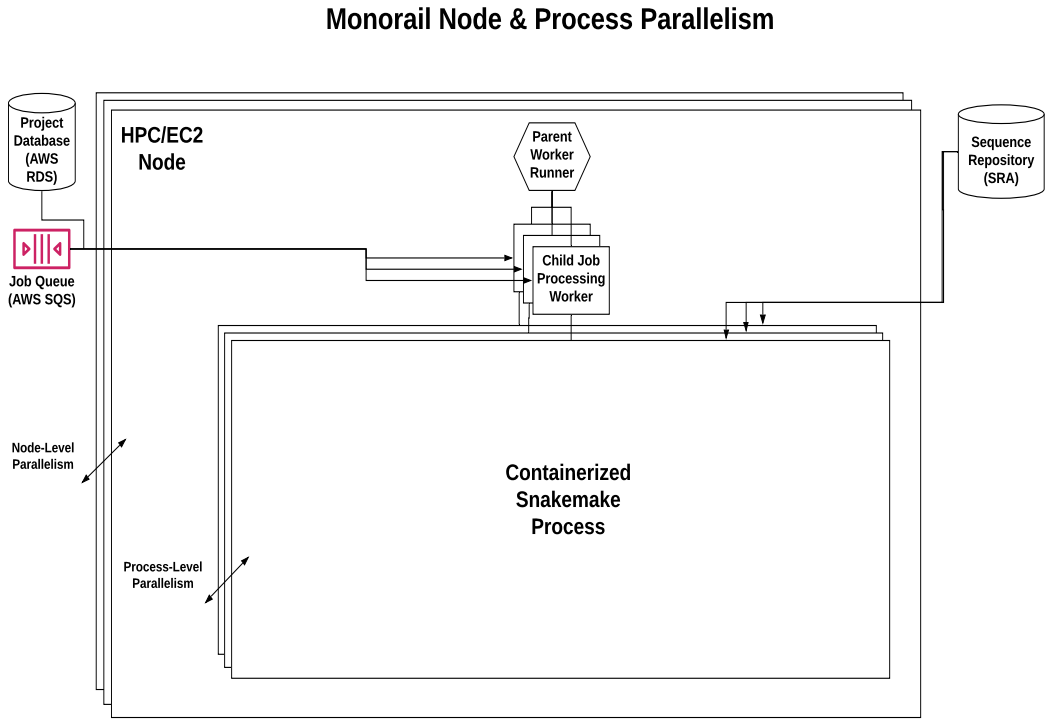

Figure 3.6: Monorail Workflow Parallelism.

Table 3.2: Monorail performance metrics run on TACC, AWS and MARCC (approximate). Statistics for GTEx and TCGA were extrapolated from a subset of each project (9277, 1567 samples respectively). GTEx output was increased by keeping whole BAM files for a subset of the samples.

Metric	Human SRA TACC	Human SRA AWS	Mouse SRA TACC	Mouse SRA AWS	Human GTEx MARCC	Human TCGA MARCC	Totals
Sequencing Runs Processed	286,000	27,618	321,000	109,889	19,214	11,348	774,644
Compressed input size (TBs)	441.78	44.2	254.27	111.873	81	75	1,008.123
Compressed output size (TBs)	64.81	6.5	39.7	16.7	11.6	7.0	146.31
Node hours (NHs)	10,133	798	8,179	5,967	2421	1467	28,965
NHs per sequencing run	0.035	0.029	0.025	0.054	0.126	0.129	0.066
NHs per compressed input TB	22.9	18.1	32.2	53.3	29.9	19.6	29.3
Sequencing runs per NH	28	35	39	18	8	8	23
Compressed input TB per NH	0.044	0.055	0.031	0.019	0.033	0.051	0.039

estimate this would cost about $0.033 per accession using equivalent cloud resources, improving substantially on the $0.93 per accession achieved by our previous Rail-RNA system (Nellore et al., 2016a).

Roughly speaking, this cost is higher but within a factor of about four times the per-accession costs of other large-scale analysis pipelines (Ziemann, Kaspi, and El-Osta, 2019; Lachmann et al., 2018). The difference is due to the fact that Monorail produces more outputs – e.g. unannotated splicing and coverage-level summaries – and relies less on gene annotation. The other systems can be more costly in the long run since a change of gene annotation requires a full re-run of the workflow. Monorail, by contrast, can quantify a new gene annotation directly from the per-base coverage files, bypassing the costlier components of the full workflow.

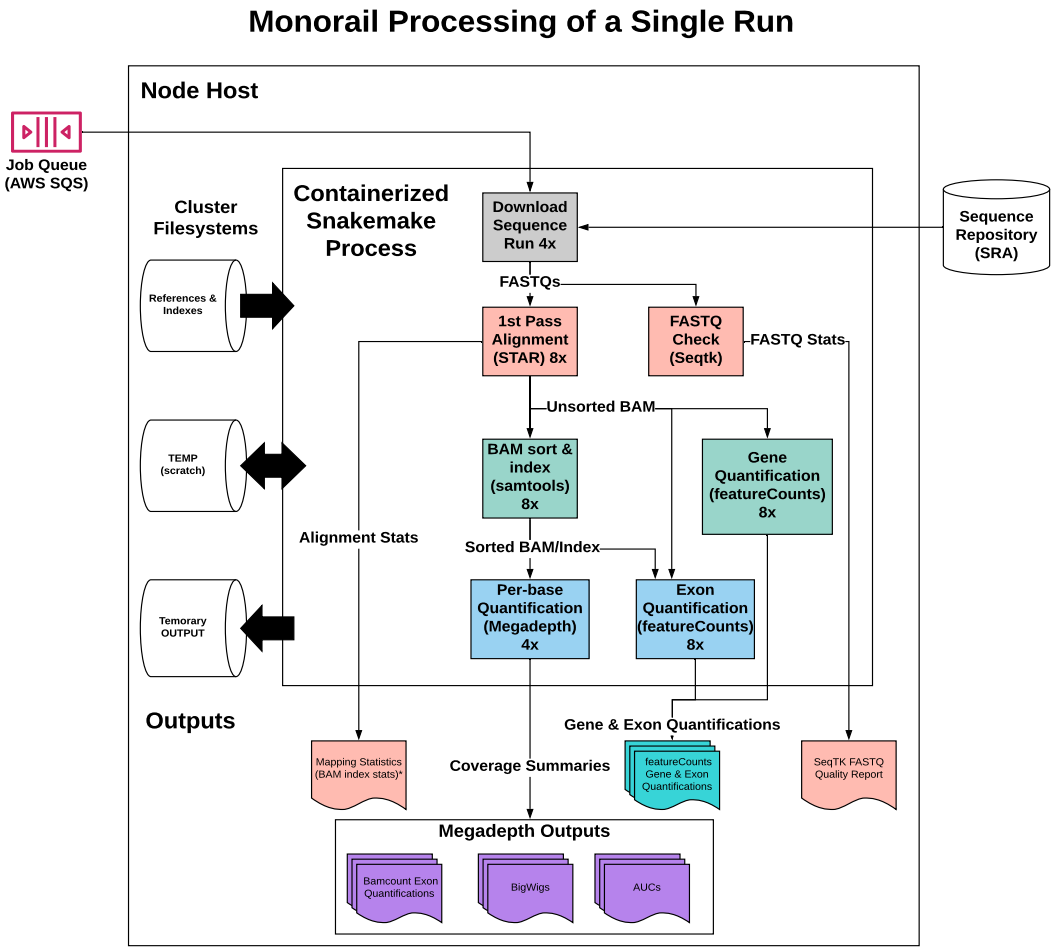

Figure 3.7: Monorail Workflow Details

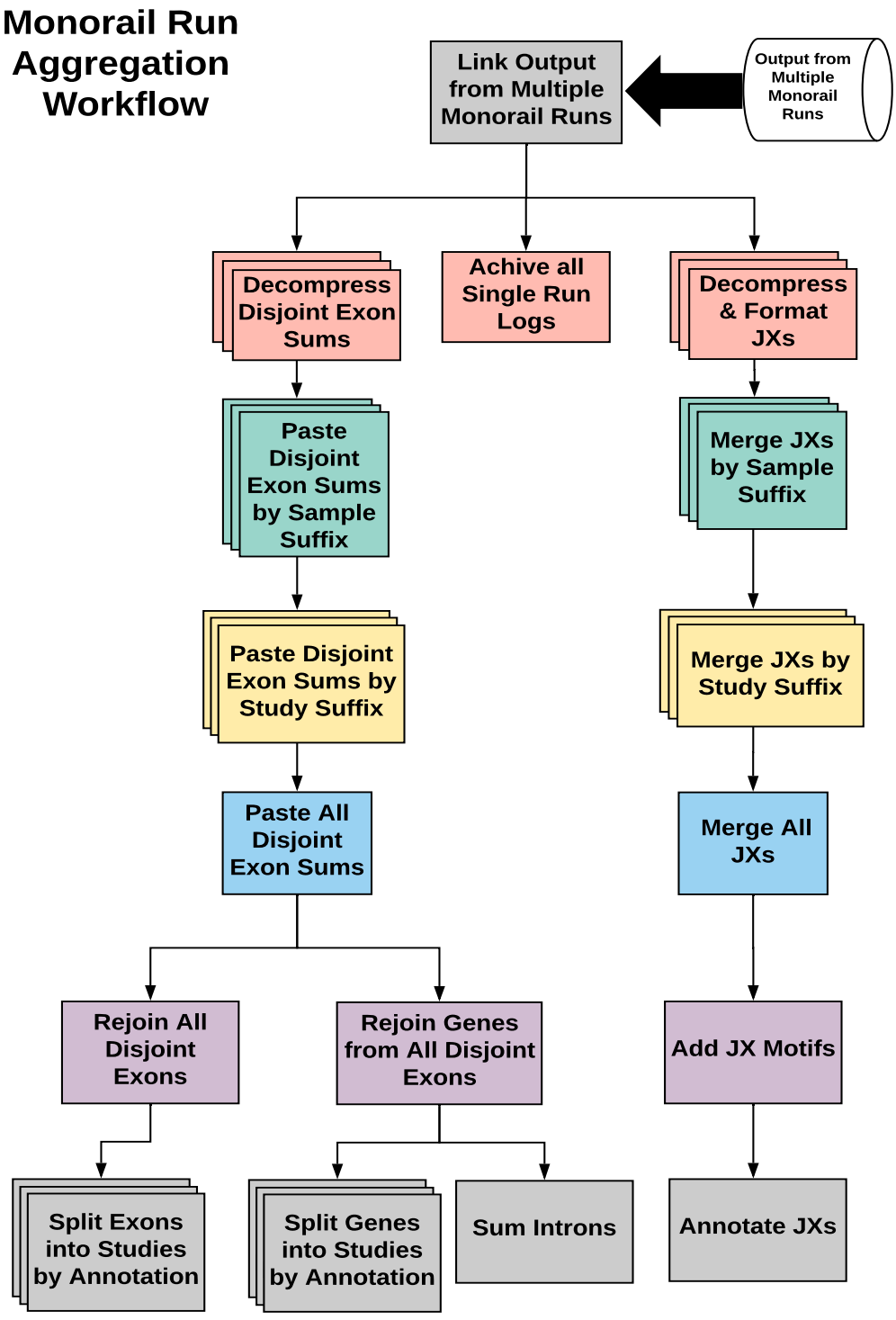

Figure 3.8: Monorail Aggregation Workflow

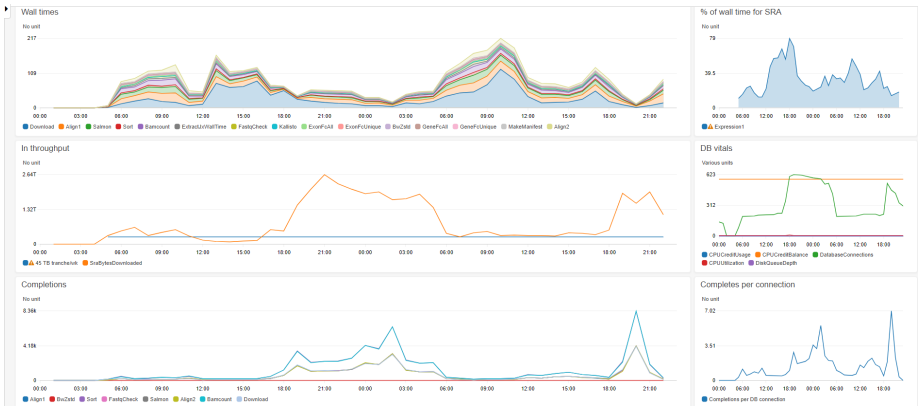

Figure 3.9: Screen shot of the Monorail monitoring interface hosted on Amazon Web Services. It uses the AWS CloudWatch Dashboards feature to allow us to monitor the performance of the Monorail system in real time. Shown are just six of the many metrics that we track.

3.5.3 Data Presentation

3.5.3.1 Snaptron

While recount offers the user a way of accessing gene, exon, and junction coverage, it is limited to providing that only at the study level. Snaptron (Wilks et al., 2018) and its newly added R interface, Snapcount, provide the ability to query precise regions of the genome for the coverage generated in Monorail. Queries can be made for a specific subset of samples (or all of them) at the gene, exon, and junction level. Queries can be further filtered by aggregate sample occurrence and read coverage. Additionally, these tools enable “higher-level" analyses to be carried out across region queries to support operations such as percent spliced in (PSI) and tissue specificity (in the case of GTEx). Snapcount specifically creates filtered RangedSummarizedExperiments dynamically based on the user’s query in contrast to the fixed nature of recount3’s study-level data objects.

Chapter 4

LongTron: Automated Analysis of Long Read Spliced Alignment Accuracy

4.1 Introduction

The last two chapters have dealt exclusively with the output of short-read RNA sequencing. In contrast with short-read RNA sequencing, long read RNA sequencing is comparatively recent and has the capacity to complement or even surpass short read RNA-seq in its ability to span several exons and splice junctions of an isoform. This capability can in principle aid the discovery of novel isoforms and the expression of existing isoforms in specific tissues and cell types (Figure 4.1). However, high error rates and other problems are still very present in the nascent field of long-read RNA sequencing and the tools that work on long reads. In this chapter we analyze the failure modes of spliced alignment of both Oxford Nanopore and PacBio Single Molecule, Real-Time (SMRT) long reads when aligned with the popular Minimap2 spliced-aligner (Li, 2018).

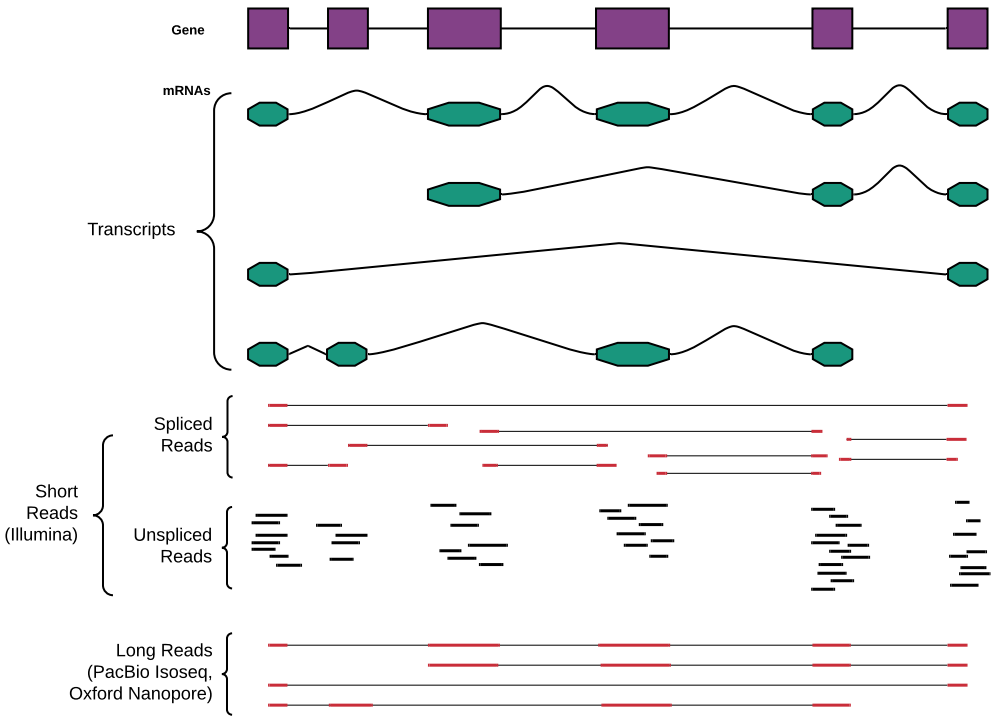

Figure 4.1: Long-read versus short-reads. While short reads have much lower error rates (1% vs. 10%) and higher coverage they lack the general ability to connect multiple splicing interactions across the transcript due to their extreme shortness (250 bases vs. 10K's bases).

As promising as they may be for transcriptomics, long reads have a few problems currently including higher error rate, higher cost per read, and potential 3-prime end bias compared with short-reads (Mantere, Kersten, and Hoischen, 2019; Amarasinghe et al., 2020). A fundamental challenge is that long reads suffer from a much higher error rate (2-10%) that is less systematic than the lower error rate of Illumina short reads (<1%, mostly toward the 3' end of the read). Additionally, the likely lower throughput due to higher cost per read of long reads, may make transcript quantification and differential analyses more difficult across the transcriptome due to lower coverage at any given locus if whole transcriptome analyses are desired (Kovaka et al., 2019).

These problems have negative effects on downstream efforts aimed at understanding transcription, including the first step in most analyses, alignment. Recently, the multi-mode aligner, minimap2 (Li, 2018) was released and is gaining popularity in long read related work, both in DNA and RNA contexts. Minimap2 is fast and relatively accurate and these authors support its continued use. However, no aligner is perfect, and minimap2 does make mistakes, specifically in the areas of spliced-alignment and the mapping of long reads' ends.

Thus this work is an initial attempt at studying and elucidating the cases where alignments of spliced long reads, both from PacBio IsoSeq and Oxford Nanopore DirectRNA, break down.The rest of the paper is divided into sections covering (1) the simulation of long reads and their alignments for benchmarking, (2) the random forest approach we took to predicting error categories of aligned long reads on both simulated and real datasets, and (3)

the concordance between long reads and short reads with respect to individual splice junctions and whole transcripts, and (4) the results of our prediction approach run on real datasets.

4.2 Related Work

This work extends the Qtip algorithm (Langmead, 2017) that also attempted to profile alignment quality/errors using a Random Forest. Where LongTron primarily differs is that we focus on the spliced alignment of long reads using minimap2 whereas Qtip focused on unspliced alignment of DNA short reads using Bowtie2 (Langmead and Salzberg, 2012), BWA-mem (Li, 2013), and SNAP (Zaharia et al., 2011) aligners.

Another related work is the FLAIR pipeline (Tang et al., 2020) which seeks to improve the spliced alignment of long reads. We utilized the FLAIR pipeline in our comparisons with raw minimap2 alignments in the results section of this paper. FLAIR uses known splice junctions from annotation and short read sequencing to correct and filter the set of spliced alignments for long reads. While FLAIR is a useful tool for correcting and refining the set of alignments, its use of annotated splice junctions makes it potentially problematic for studies looking for novel splicing in long read alignments. A related pipeline similarly profiling long reads, specifically for the PacBio platform is SQANTI (Tardaguila et al., 2018). SQUANTI and its successor SQUANTI2 (https://github.com/Magdoll/SQANTI2/) are intended to classify PacBio long reads spliced alignments and also use a Random Forest to classify artifactual results.

4.3 Methods

4.3.1 Long read failure modes

Typically RNA-seq aligners leverage heuristics to find a set of near-optimal candidate locations in the genome for the placement of both short and long reads. For RNA sequence analysis these heuristics are particularly relevant for at least two phases of the alignment process, commonly called seed-and-extend. In the first phase, the alignment search space is narrowed down from being the full genome to a short list of candidate loci (using seeds). These seeds are chosen in different ways by various aligners, although they often use heuristics that don't guarantee an optimal alignment will always be identified (Darby et al., 2020). In the second phase, candidate loci are more thoroughly checked for their compatibility with the query sequence which includes splice-site determination. The Smith-Waterman optimal algorithm (Smith and Waterman, 1981) for local sequence alignment can be used efficiently at this stage to produce a gapped alignment. However, this is not useful for spliced alignments which still require a heuristic to determine the best splice-sites around which to split the query sequence.

4.3.2 Long Read Transcriptome Simulation

To assess the accuracy of a long read RNA-seq analysis pipeline, we first used a simulation approach so that we could precisely measure the alignment accuracy and splicing results of the simulated reads compared to their ground truth. For this, we started with the Gencode version 28 annotation and the

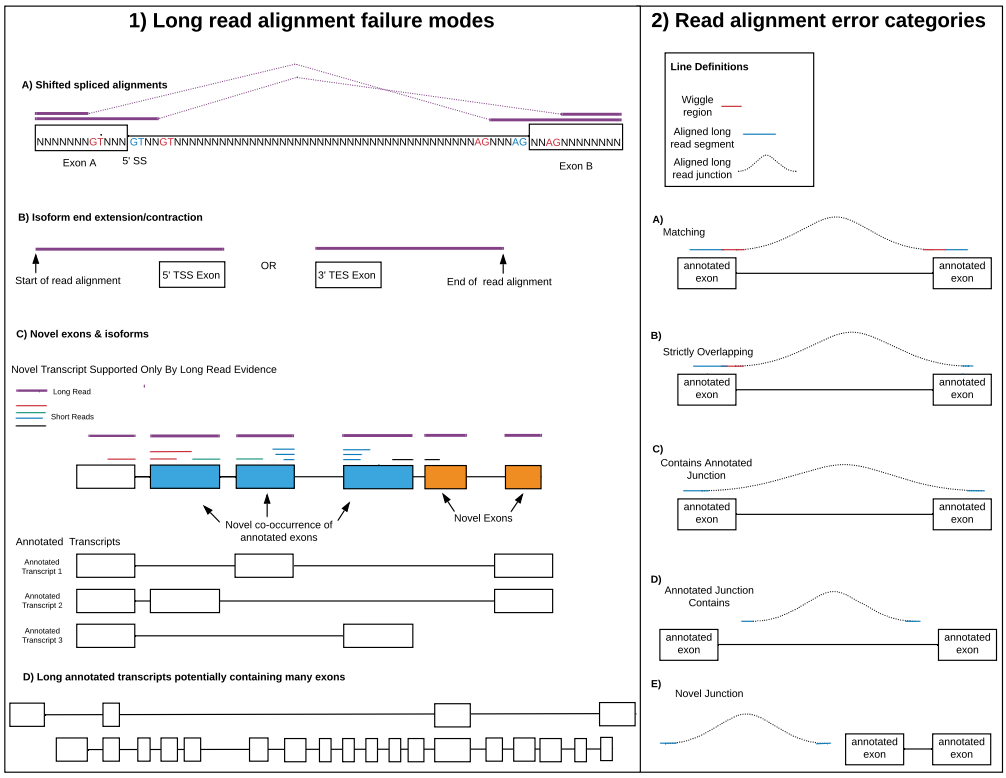

Figure 4.2: 1. Long read alignment failure modes. A) Spliced alignments can shift in the presence of unannotated splice motifs in the reference near annotated (real) splice sites. B) 5′ and 3′ ends of isoforms are difficult to get right as sequencing the ends of long reads is imprecise. C) Long reads can produce novel configurations of annotated exons and/or novel exons. However, these may be simply alignment artifacts due to splice motifs and/or repeats in the region (e.g. the rightmost novel exon has no short read support). D) Large numbers of exons (splice sites) can result in multiple novel long read alignments, some of which may be false. This is in part due to the non-full length nature of many of the long reads (especially from PacBio). 2. Read alignment error categories. A) Matching junction alignment against at least one source transcript junction; B) Alignment overlapping any transcripts' junction; C) Alignment containing any transcripts' junctions; D) One or more transcripts' junctions containing aligned junction; E) Junction is completely novel

error profile from SURVIVOR (Jeffares et al., 2017b) for both Oxford Nanopore and PacBio IsoSeq derived from minimap2 alignments of NA12878 reads to the Gencode transcriptome. The NA12878 sample is from a disease-free human and has been used by many other research efforts. Using these we simulated long reads from the transcript sequences, both full-length and partial length. We then aligned these simulated reads against the genome and extracted features from each alignment. These features were then evaluated by a random forest for training and prediction. Our implementation used the RandomForestClassifier in the scikit-learn Python machine learning framework. We used 100 trees and eight parallel threads for training. For the purposes of Receiver Operator Curve (ROC) plotting we used the predict_proba method on the held-out test set. The genomic alignments of the simulated reads were used to determine four correctness categories. We experimented with using both these four categories as a multi-class prediction problem in the random forest as well as a more simple binary model where the three non problem-free categories in the list below were collapsed into one category.

Junction alignments were first categorized into five subcategories, allowing a margin ("fuzz" or "wiggle") of up to 20 nucleotides on each end, as described in Figure 4.2-2. These five categories were then categorized into the four top-level correctness classes:

- Problem-free (A)
- Any error (alignment in any of B-D but not all three)
- Recurrent error (alignments in all three B-D)

- Novel (E)

With the exception of splice motifs as the third most important feature in the Oxford full-length run, exon length dominated the Oxford feature importance rankings (Table B.1 in Appendix B). Similarly, both exon and transcript length were among several of the top most important features for PacBio. In addition GC content was the third most important feature for the PacBio full-length run. A selection of these features are shown in Figure 4.3-2.

4.4 Results

4.4.1 Training and Application

We trained four distinct random forest models using the final set of features described above:

- PacBio IsoSeq Full Length
- PacBio IsoSeq Fragment
- Oxford Nanopore Full Length
- Oxford Nanopore Fragment

Training accuracy was high on a held out test dataset (Figures B.2, B.3, B.4, and B.5 in Appendix B).

We then applied both full- and fragment-length models to the minimap2 alignments of long reads from PacBio and Oxford sequencing of the NA12878 sample. We intersected the long reads alignments with transcripts of known

Table 4.1: Counts of alignments in each simulated training class

Dataset	Total	Problem-free	Any error	Recurrent error	Novel
Oxford full length	1,696,509	41.90% (710,869)	55.42% (940,159)	2.30% (38,970)	0.38% (6,511)
Oxford fragment	1,668,351	75.09% (1,252,754)	19.64% (327,629)	2.29% (38,252)	2.98% (49,716)
PacBio CCS full length	971,786	88.37% (858,755)	9.75% (94,708)	1.33% (12,883)	0.56% (5,440)
PacBio CCS fragment	956,945	91.98% (880,189)	6.57% (62,837)	0.72% (6,900)	0.73% (7,019)

error categories to get the ground truth. This allows us to compute a form of recall and precision of the predictions (Tables B.2 and B.3 in Appendix B).

These results show Oxford had more errors than PacBio and also full-length alignments are more difficult to achieve than are fragments (Table 4.1). The PacBio IsoSeq platform supports the ability to generate a set of higher quality long reads by continuing to sequence the same molecule iteratively in a process called Circular Consensus Sequencing (CCS) (Gordon et al., 2015). The NA12878 PacBio dataset we are using is CCS corrected which likely contributes to its higher problem-free percentages.

4.5 Splice-junction and Isoform Comparison

A significant portion of the work described here involved comparison of splicing and isoforms across both long read sequencing approaches as well as Illumina short read sequencing. In Table 4.2 we present a comparison of splice junctions between the three long read sequencing samples we used and a large compendium of putative splice junctions called from Illumina short reads, used in the Snaptron tool (Wilks et al., 2018).

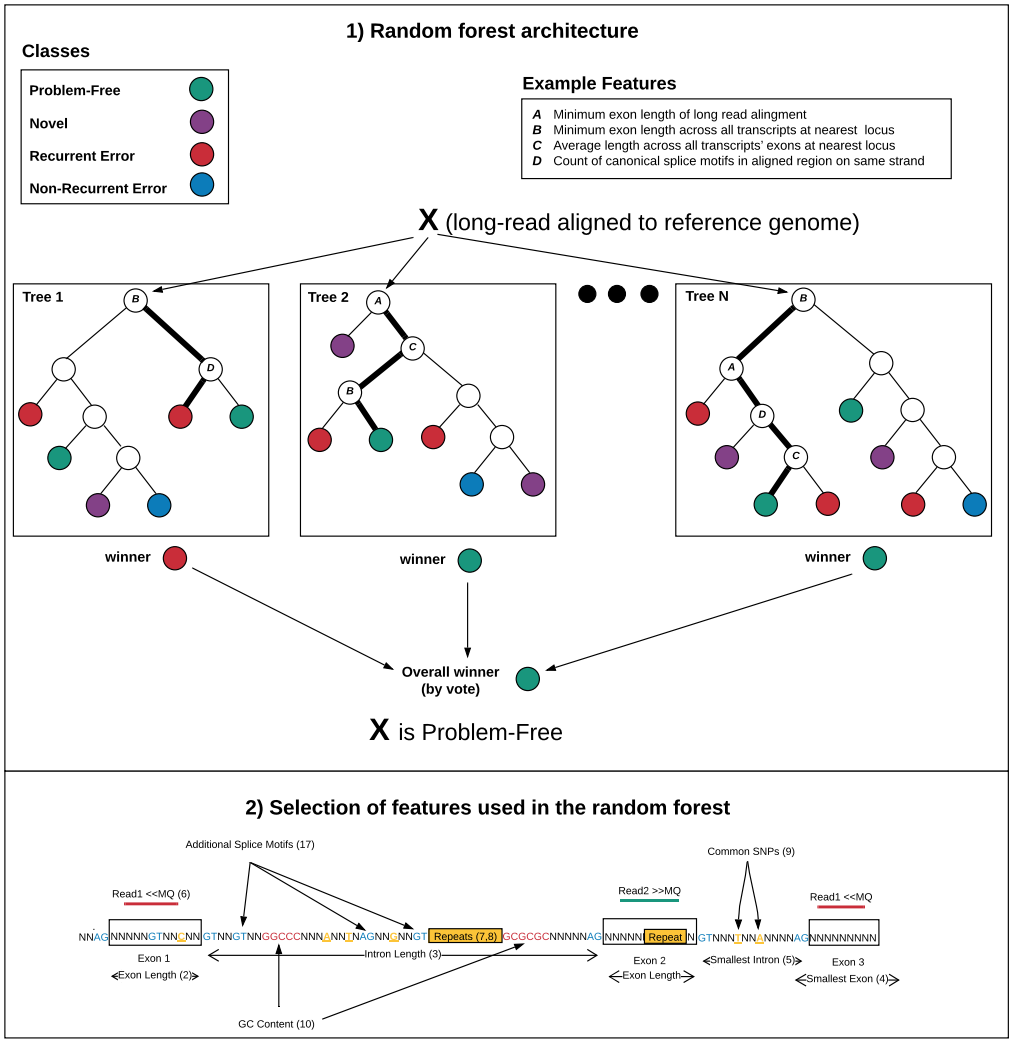

Figure 4.3: 1. Random forest classification. 2. Diagram of a selection of features used in the random forest, including 1-10 and 17 from the full category list in Appendix B

As seen in the table requiring exact matches between the aligned long reads and the short-read based splice sites results in a minority of splice junctions matching in the two annotated categories. Allowing for a "fuzz" (20 nt) on both ends of a match greatly increases concordance between long reads and short reads. This discrepancy between exact and fuzz matching, specifically with annotated junctions, highlights the difficulties in long read alignment discussed in this paper.

In contrast, the "All short-read supported exact" and "Full novel exact" categories fare considerably better in concordance. This is one benefit of using a large group of short-read-derived splice junctions which include many putative novel junctions. Most of these matches would have been missed if a pseudo-alignment/quasi-mapping strategy had been used to derive the short read junctions.

Overall, the results from splice-junction concordance is relatively positive. While long read alignments fail to pick up many annotated junctions under the strictures of exact matching, the majority of them are relatively close in terms of genomic coordinates. Further, there is evidence here to suggest long read alignments are supporting other, novel junctions previously found in short reads.

Another area of importance is isoform level concordance. We've already noted the potential benefit of long reads in finding novel isoforms where these can either use novel exons/splice junctions, or more likely, novel groupings of existing exons/splice junctions. Table 4.3 presents the isoform-level comparison results using intron-chains as a proxy for isoforms. Intron chains, as their

Table 4.2: Splice Junction Comparison (Snaptron represents a compendium of short-read derived junctions, annotated and novel), fuzz=20 for bases on either side, percents do not add up to 100 as annotated short-reads are a subset of all short-reads. Junctions are compared by coordinates alone (strand not included).

Long Read Sample	**Total**	**Gencode V29 exact** (360,700)	**Annotated short-read supported exact** (Snaptron, 445,651)	**All short-read supported exact** (111,160,460)	**Fully novel exact**	**Gencode V29 fuzz**	**All short-read supported fuzz**	**Fully novel fuzz**
PacBio NA12878	301,666 (100%)	96,124 (32%)	99,394 (33%)	115,496 (38%)	186,170 (62%)	258,094 (86%)	292,751 (97%)	8,915 (**3%**)
Oxford NA12878	612,614 (100%)	163,199 (27%)	180,028 (29%)	338,773 (55%)	273,841 (45%)	306,045 (50%)	508,798 (83%)	103,816 (**17%**)
PacBio SKBR3	1,639,588 (100%)	143,371 (9%)	152,346 (9%)	242,903 (15%)	1,396,685 (85%)	1,001,378 (61%)	1,271,853 (78%)	367,735 (**22%**)
Gencode V29	360,700 (100%)	NA	NA	346,810 (96%)	NA	NA	350,671 (97%)	10,029 (**3%**)

name implies, restrict comparison to the order and identity of the genomic coordinates which make up the donor/acceptor sites within the isoform. Thus start/end coordinates of the isoform as a whole are ignored. This will miss differences arising from alternative transcript start/end sites although these are intrinsically the most difficult to sequence because of the protocols involved (Workman et al., 2019) and (Roach et al., 2020).

The totals column in Table 4.3 represents deduplicated sets of intron chains (additional details in section 2 and Table B.4 in Appendix B). The intron-chains between samples show little concordance when exact matching is required. This phenomenon is far worse than in the splice-junction level analysis (Table 4.2), even when compared with counts of exact matches of splice-junctions. These initial results spurred us to involve an additional approach to use in filtering, the FLAIR (Tang et al., 2020) pipeline. It also required us to modify an existing tool, gffcompare (Pertea and Pertea, 2020), to allow for fuzz when comparing intron chains.

Table 4.3: Isoform comparison table, using gene models from Gencode V29, plus the isoforms from all the union of annotations; both exact and fuzz comparisons of the set of long-read derived isoforms which 1) match in number of introns or 2) are contained or contain a reference isoform.

Sample	Total Intron Chains	Gencode V29 (199381)	Union of Annotations (1098511)	Short Assembly	(Other) Long Reads
PacBio SKBR3 (PB-SKBR3)	1,339,872	22.0% (60.2%)	27.7% (63.0%)	Illumina: 16.7% (44.2%)	NA
PacBio NA12878 FLAIR	13,026	87.1% (95.6%)	93.3% (98.1%)	Illumina: 76.6% (86.9%)	OX-FLAIR: 86.4% (92.3%), PB-RAW: 93.5% (97.1%)
PacBio NA12878 (PB-RAW)	516,021	41.4% (81.0%)	43.4% (76.9%)	Illumina: 39.0% **(64.6%)**	OX-RAW: **61.4% (94.6%),** PB-FLAIR: 38.6% (75.1%)
Oxford NA12878 FLAIR	43,817	65.7% (86.5%)	79.8% (93.3%)	Illumina: 42.4% (62.5%)	OX-RAW: 98.5% (100.0%), PB-FLAIR: 30.1% (42.7%)
Oxford NA12878 (OX-RAW)	6,744,568	73.2% (89.1%)	77.2% (90.9%)	Illumina: 67.4% **(83.5%)**	OX-FLAIR: 73.4% (87.8%), **PB-RAW: 87.6% (94.5%)**
Illumina SKBR3	30,866	63.2% (81.0%)	77.6% (90.1%)	NA	PB-SKBR3: 69.9% (81.4%)
Illumina NA12878	101,151	37.5% (57.6%)	56.7% (73.5%)	NA	OX-RAW: 63.9% **(72.3%)**, OX-FLAIR: 21.9% (33.2%), **PB-RAW:** 39.0% **(46.8%)**, PB-FLAIR: 16.4% (21.4%)

One issue we encountered was the ambiguity of strand of origin for the PacBio long reads. We noticed that a large number of mismatching PacBio read alignments were classified as matching but on the opposite strand when compared with the "union of annotations" transcript set. By considering the PacBio alignments which were classified by gffcompare as opposite strand matches (categories "o" and "s") and swapping their strands, and then re-comparing, a larger number of alignments were correctly re-classified for PacBio. In contrast, changing the strand parameter ("-u") in minimap2 had little effect. This is an important issue to consider when aligning PacBio-derived long reads with minimap2.

A key finding in Table 4.3 is that allowing for fuzz around junction boundaries makes a substantial contribution to raising the number of matching

intron chains across almost every category. This again underscores one of the key problems in long read alignment, that is any difficulties computing the exact coordinates of a single junction correct are magnified when chaining together multiple of those junctions into isoforms.

However, even without fuzz, both Oxford (NA12878) and PacBio (SKBR3) aligned samples are able to capture a larger amount of the annotated intron chains than their short read assembled counterparts (Illumina NA12878/SKBR3). The fact that the PacBio NA12878 sample falls behind here may be due to the much lower numbers of reads present in that sample. This bodes well for long read sequencing in the future in terms of finding coverage for annotated isoforms. In addition, the FLAIR pipeline raises concordance dramatically but at the cost of a substantial reduction in total isoforms.

Further, even when requiring exact junction coordinate matches, the concordance between Oxford and PacBio is fairly high (61.4% and 87.6% respectively), while with a fuzz of 20bp the numbers both jump to 94% of each set. The lower percent of Oxford captured by PacBio is most likely due to the much smaller size of the PacBio read set. This is also probably the explanation for the lower percentage of Illumina-assembled intron chains captured by the NA12878 PacBio (46.8%) even with fuzz, while the short read assembly is capturing a majority of the PacBio long read intron chains (64.6%) with fuzz. Oxford in comparison is both capturing and being captured at a high rate by the Illumina assemblies (83.5% and 72.3% with fuzz, respectively).

4.5.1 Effects of Random Forest Classifier on Transcript Matching against the Annotation

We further took the set of NA12878 (Pacbio and Nanopore) and SKBR3 alignments predicted to be in the "problem-free" category and used them in comparisons against the "union of annotation" transcript set to see if the number of matching intron-chains improved (Table B.4). While this strategy improved the precision—the percent of total query read alignments which matched an annotated transcript (NA12878 or SKBR3)—it substantially lowered the recall—the percent of annotated transcripts matching query read alignments. This was in large part due to using the set requiring both the full-length and the fragment models to predict a problem-free alignment. Recomputing the comparison with the union of full-length and fragment models' predictions results in close to the original recall while only slightly lowering the improved precision for Nanopore, but with less impact on PacBio.

4.5.2 Novel Alignment Examples in NA1878 and SKBR3

We next evaluated the use of long read RNA sequencing to discover novel (unannotated) transcripts in the genome (Figures 4.4 and 4.5). In Figure 4.4, both Oxford and PacBio long-reads from the NA12878 sample support some additional transcription before the start of the NPIPB5 gene. This could be a novel alternative transcription start site (TSS). The far reduced support from PacBio reads could be a factor due to the much smaller total read set in that sequencing experiment. Figure 4.5 displays a region of potential novel transcription found primarily in the SKBR3 PacBio long-read sample. While

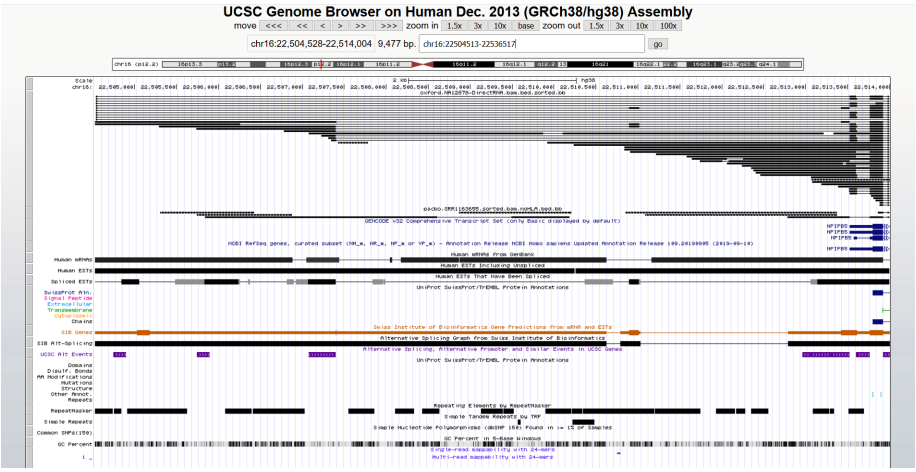

Figure 4.4: Novel transcript predicted region on NA12878 for both Oxford and PacBio

a small subset of the region has minor support in the Oxford sequenced NA12878 sample, the majority of the transcription appears to be exclusive to SKBR3. There appears to be further evidence from human mRNA/ESTs that this region is indeed transcribed and is not due to technical error

4.6 Discussion

Long reads are useful for finding new isoforms as combinations of splice junctions that have already been found by short reads, but caution must be exercised due to the failure modes described here. Our investigation will help in assessing long read alignments to make more confident calls as to 1) errors and 2) novel cases.

While there are a number of potential factors influencing how long reads are aligned, based on our investigation and the results of our random forest experiments, a few rise to the top in terms of importance. An important

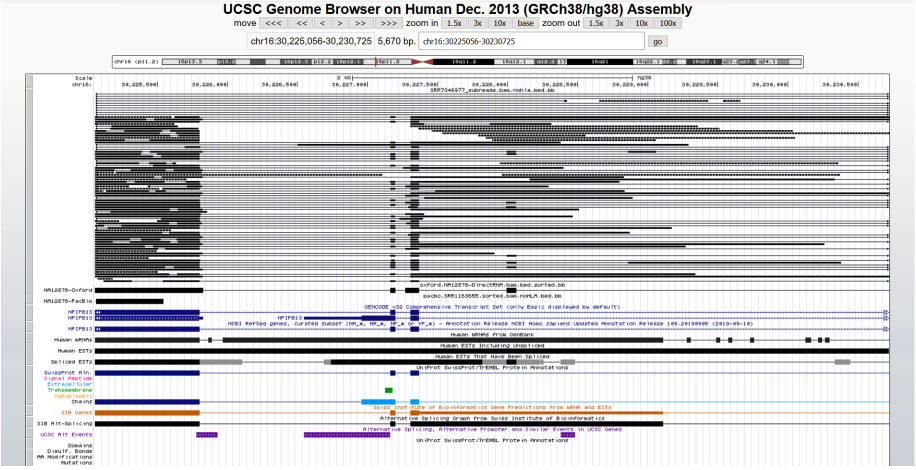

Figure 4.5: Novel transcript predicted region on SKBR3 PacBio

factor is the number of exons present in a gene. If there are more exons in an isoform, then that translates into a larger number of potential splice-site determination errors the aligner can make when aligning long reads which often are still fragments of the full length isoform. A related factor is the number of alternative isoforms present in the gene. This also raises the potential for splice-site finding errors in the aligner as many exons may be shared, while others may overlap but with different starts/ends, while still others are completely novel. This can lead to long reads missing certain alternative splice sites while supporting others within the same gene.

Further, the significant decrease in coverage within the currently available long-read sequencing datasets substantially reduces confidence in putative novel regions. This is partly alleviated by short reads, at least for single exon genes and combinations of a few splice junctions. However, long novel transcripts combining many exons in new ways will be harder to substantiate. It's also important to consider that nucleotide sequence alignment in general is

almost always heuristic-based, and this is certainly true of spliced-alignment. While better alignment heuristics, modeling, and short-read sequencing may be able to fill in some of the gaps left in long-read spliced-alignments, ultimately there will need to be either a significant decrease in error rates or a substantial increase in coverage to alleviate at least some of the problems reported here.

Chapter 5

Discussion and Conclusion

We have presented several tools, and the RNA-seq data processed through those tools, to aid the downstream biomedical researcher in the process of discovering, ranking, and validating splicing-related hypotheses (Snaptron, LongTron), as well as more general gene expression related questions (Monorail, recount3, and Megadepth). As part of that presentation, we have demonstrated a nascent genomic search engine specifically with the Monorail and Snaptron projects serving the roles of the "sequence crawler" and query interface, respectively. Further, we have analyzed the error types that can arise from long-read spliced-alignments using Minimap2 and the overall concordance of splicing calls in long reads compared against short reads and annotation, in the LongTron project.

Much work remains in all of these areas. RNA-seq is simply the starting place as researchers must be able to take advantage of the multiple sequencing and related technologies available today beyond RNA-seq (bisulfite sequencing to capture methylation, ChIP-seq, proteomics, and DNA whole genome sequencing). Even limiting the focus to just RNA-related technologies, there

still remains the ever growing compendium of single-cell RNA-seq outside of the bulk and smart-seq samples captured here, much of which comes from the 10x Chromium platform (Zheng et al., 2017). In the latter case the Monorail workflow must be adapted to handle the variety of formatting used to represent the additional information needed to appropriately process single-cell RNA reads (cell-barcodes and universal molecular identifiers "UMIs"). Ideally all the various technologies would be integrated into a single, uniform resource harmonized as much as possible. This is likely an ambitious goal, but the recount3 and Snaptron2 resources described in chapter 3 serve as an excellent foundation to work from.

A second, but very related, direction for additional work is the area of sequencing and sample metadata. The value of the existing primary data in the public repositories could be increased through better metadata curation, specifically in the areas of tissue, cell-type, and disease annotation for consistency and completeness across the SRA. Without reliable and consistently applied metadata of this type, the usefulness of the primary sequencing data and the summaries derived from it, is severely limited. While manual curation is one approach, it fails to scale to the 100,000's of sequencing runs present in the SRA and could also introduce additional inconsistencies. A potentially better approach, both for its scalability and consistency, is to use the primary data itself to inform the labeling of tissues, cell-types, and other important fields. Such an approach has already seen some success (Ellis et al., 2018b) but needs to be updated for human data and extended to other organisms (e.g. mouse). Related to this, an immediately useful contribution would be

identifying one or a few high quality "reference" RNA-seq studies that could serve as GTEx-like transcriptomics standards in mouse. These studies would need to have both consistently processed primary sequencing data as well relatively complete and consistent metadata.

Ultimately, resources such as recount3 and Snaptron are force-multipliers that can take the efforts of a few and dramatically increase the potential output of the many. This justifies the relative high initial cost of investment in large, complex systems such as Rail and Monorail and the ongoing maintenance they entail. However, it also highlights the still serious burden of what should by now be trivial steps in the analysis pipeline, specifically the transfer and alignment operations. This burden also highlights the complexities of biology, a fact that these analyses must ever take into consideration. It remains to be seen, especially in the current era of sophisticated machine learning, whether the mysteries held in biological data will be as easily elucidated as other data types have been.

Bibliography

Langmead, B. and A. Nellore (2018). "Cloud computing for genomic data analysis and collaboration". In: *Nat. Rev. Genet.* 19.4, pp. 208–219.

Dijk, E. L. van, H. Auger, Y. Jaszczyszyn, and C. Thermes (2014). "Ten years of next-generation sequencing technology". In: *Trends Genet.* 30.9, pp. 418–426.

Bray, N. L., H. Pimentel, P. Melsted, and L. Pachter (2016a). "Near-optimal probabilistic RNA-seq quantification". In: *Nat. Biotechnol.* 34.5, pp. 525–527.

Patro, R., G. Duggal, M. I. Love, R. A. Irizarry, and C. Kingsford (2017). "Salmon provides fast and bias-aware quantification of transcript expression". In: *Nat. Methods* 14.4, pp. 417–419.

Dobin, A., C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras (2013). "STAR: ultrafast universal RNA-seq aligner". In: *Bioinformatics* 29.1, pp. 15–21.

Goldstein, L. D., Y. Cao, G. Pau, M. Lawrence, T. D. Wu, S. Seshagiri, and R. Gentleman (2016a). "Prediction and Quantification of Splice Events from RNA-Seq Data". In: *PLoS ONE* 11.5, e0156132.

Haas, B. J., A. Dobin, B. Li, N. Stransky, N. Pochet, and A. Regev (2019). "Accuracy assessment of fusion transcript detection via read-mapping and de novo fusion transcript assembly-based methods". In: *Genome Biol.* 20.1, p. 213.

Trapnell, C., B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter (2010). "Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation". In: *Nat. Biotechnol.* 28.5, pp. 511–515.

Pertea, M., G. M. Pertea, C. M. Antonescu, T. C. Chang, J. T. Mendell, and S. L. Salzberg (2015). "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads". In: *Nat. Biotechnol.* 33.3, pp. 290–295.

Nellore, A., L. Collado-Torres, A. E. Jaffe, J. Alquicira-Hernandez, C. Wilks, J. Pritt, J. Morton, J. T. Leek, and B. Langmead (2016a). "Rail-RNA: scalable analysis of RNA-seq splicing and coverage". In: *Bioinformatics*.

Zhang, D., S. Guelfi, S. Garcia-Ruiz, B. Costa, R.H. Reynolds, K. D'Sa, W. Liu, T. Courtin, A. Peterson, A.E. Jaffe, et al. (2020). "Incomplete annotation has a disproportionate impact on our understanding of Mendelian and complex neurogenetic disorders". In: *Science Advances* 6.24, eaay8299.

Pertea, M., A. Shumate, G. Pertea, A. Varabyou, F.P. Breitwieser, Y. Chang, A.K. Madugundu, A. Pandey, and S.L. Salzberg (2018). "CHESS: a new human gene catalog curated from thousands of large-scale RNA sequencing experiments reveals extensive transcriptional noise". In: *Genome biology* 19.1, pp. 1–14.

Collado-Torres, L., A. Nellore, K. Kammers, S. E. Ellis, M. A. Taub, K. D. Hansen, A. E. Jaffe, B. Langmead, and J. T. Leek (2017b). "Reproducible RNA-seq analysis using recount2". In: *Nat. Biotechnol.* 35.4, pp. 319–321.

Nellore, A., C. Wilks, K. D. Hansen, J. T. Leek, and B. Langmead (2016c). "Rail-dbGaP: analyzing dbGaP-protected data in the cloud with Amazon Elastic MapReduce". In: *Bioinformatics* 32.16, pp. 2551–2553.

Nellore, A., A. E. Jaffe, J. P. Fortin, J. Alquicira-Hernandez, L. Collado-Torres, S. Wang, R. A. Phillips Iii, N. Karbhari, K. D. Hansen, B. Langmead, and J. T. Leek (2016b). "Human splicing diversity and the extent of unannotated splice junctions across human RNA-seq samples on the Sequence Read Archive". In: *Genome Biology* 17.1, p. 266.

Razmara, Ashkaun, Shannon E Ellis, Dustin J Sokolowski, Sean Davis, Michael D Wilson, Jeffrey T Leek, Andrew E Jaffe, and Leonardo Collado-Torres (2019). "recount-brain: a curated repository of human brain RNA-seq datasets metadata". In: *BioRxiv*, p. 618025.

Imada, E. L., D. F. Sanchez, L. Collado-Torres, C. Wilks, T. Matam, W. Dinalankara, A. Stupnikov, F. P. Pereira Lobo, C. W. Yip, K. Yasuzawa, N. Kondo, M. Itoh, H. Suzuki, T. Kasukawa, C. C. Hon, M. J. de Hoon, J. W. Shin, P. Carninci, A. E. Jaffe, J. T. Leek, A. Favorov, G. R. Franco, B. Langmead, and L. Marchionni (2020). "Recounting the FANTOM CAGE-Associated Transcriptome". In: *Genome Res.*

Dean, Jeffrey and Sanjay Ghemawat (2010). "MapReduce: a flexible data processing tool". In: *Communications of the ACM* 53.1, pp. 72–77.

Pohl, A. and M. Beato (2014). "bwtool: a tool for bigWig files". In: *Bioinformatics* 30.11, pp. 1618–1619.

Sveen, A., S. Kilpinen, A. Ruusulehto, R. A. Lothe, and R. I. Skotheim (2016). "Aberrant RNA splicing in cancer; expression changes and driver mutations of splicing factor genes". In: *Oncogene* 35.19, pp. 2413–2427.

Sibley, C. R., L. Blazquez, and J. Ule (2016). "Lessons from non-canonical splicing". In: *Nat. Rev. Genet.* 17.7, pp. 407–421.

Li, H. (2018). "Minimap2: pairwise alignment for nucleotide sequences". In: *Bioinformatics* 34.18, pp. 3094–3100.

Jeffares, D. C., C. Jolly, M. Hoti, D. Speed, L. Shaw, C. Rallis, F. Balloux, C. Dessimoz, J. B?hler, and F. J. Sedlazeck (2017a). "Transient structural variations have strong effects on quantitative traits and reproductive isolation in fission yeast". In: *Nat Commun* 8, p. 14061.

Pertea, G. and M. Pertea (2020). "GFF Utilities: GffRead and GffCompare". In: *F1000Res* 9, p. 304.

Paila, Umadevi, Brad A Chapman, Rory Kirchner, and Aaron R Quinlan (2013). "GEMINI: integrative exploration of genetic variation and genome annotations". In: *PLoS Comput Biol* 9.7, e1003153.

Layer, Ryan M, Neil Kindlon, Konrad J Karczewski, Aaron R Quinlan, Exome Aggregation Consortium, et al. (2016). "Efficient genotype compression and analysis of large genetic-variation data sets". In: *Nature methods* 13.1, pp. 63–65.

Li, Heng (2016). "BGT: efficient and flexible genotype query across many samples". In: *Bioinformatics* 32.4, pp. 590–592.

Durbin, Richard (2014). "Efficient haplotype matching and storage using the positional Burrows–Wheeler transform (PBWT)". In: *Bioinformatics* 30.9, pp. 1266–1272.

Solomon, Brad and Carl Kingsford (2016). "Fast search of thousands of short-read sequencing experiments". In: *Nature biotechnology* 34.3, pp. 300–302.

Petryszak, R., M. Keays, Y. A. Tang, N. A. Fonseca, E. Barrera, T. Burdett, A. Fullgrabe, A. M. Fuentes, S. Jupp, S. Koskinen, O. Mannion, L. Huerta, K. Megy, C. Snow, E. Williams, M. Barzine, E. Hastings, H. Weisser, J. Wright, P. Jaiswal, W. Huber, J. Choudhary, H. E. Parkinson, and A. Brazma (2016). "Expression Atlas update–an integrated database of gene and protein expression in humans, animals and plants". In: *Nucleic Acids Res.* 44.D1, pp. D746–752.

Kolesnikov, Nikolay, Emma Hastings, Maria Keays, Olga Melnichuk, Y Amy Tang, Eleanor Williams, Miroslaw Dylag, Natalja Kurbatova, Marco Brandizi, Tony Burdett, et al. (2014). "ArrayExpress update—simplifying data submissions". In: *Nucleic acids research*, gku1057.

Quinlan, Aaron R and Ira M Hall (2010). "BEDTools: a flexible suite of utilities for comparing genomic features". In: *Bioinformatics* 26.6, pp. 841–842.

Lawrence, Michael, Wolfgang Huber, Hervé Pages, Patrick Aboyoun, Marc Carlson, Robert Gentleman, Martin T Morgan, and Vincent J Carey (2013). "Software for computing and annotating genomic ranges". In: *PLoS Comput Biol* 9.8, e1003118.

Kozanitis, C., A. Heiberg, G. Varghese, and V. Bafna (2014). "Using Genome Query Language to uncover genetic variation". In: *Bioinformatics* 30.1, pp. 1–8.

Guðbjartsson, Hákon, Guðmundur Fr Georgsson, Sigurjón A Guðjónsson, Ragnar Þór Valdimarsson, Jóhann H Sigurðsson, Sigmar K Stefánsson, Gísli Másson, Gísli Magnússon, Vilmundur Pálmason, and Kári Stefánsson (2016). "GORpipe: a query tool for working with sequence data based on a Genomic Ordered Relational (GOR) architecture". In: *Bioinformatics*, btw199.

Nellore, Abhinav, Andrew E. Jaffe, Jean-Philippe Fortin, José Alquicira-Hernández, Leonardo Collado-Torres, Siruo Wang, Robert A. Phillips III, Nishika Karbhari, Kasper D. Hansen, Ben Langmead, and Jeffrey T. Leek (2016d). "Human splicing diversity and the extent of unannotated splice junctions across human RNA-seq samples on the Sequence Read Archive". In: *Genome Biology* 17.1, p. 266. ISSN: 1474-760X. DOI: 10.1186/s13059-016-1118-6. URL: http://dx.doi.org/10.1186/s13059-016-1118-6.

Nellore, Abhinav, Christopher Wilks, Kasper D Hansen, Jeffrey T Leek, and Ben Langmead (2015). "Rail-dbGaP: a protocol and tool for analyzing protected genomic data in a commercial cloud". In: *bioRxiv*, p. 035287.

Collado-Torres, Leonardo, Abhinav Nellore, Kai Kammers, Shannon E Ellis, Margaret A Taub, Kasper D Hansen, Andrew E Jaffe, Ben Langmead, and Jeffrey Leek (2016). "recount: A large-scale resource of analysis-ready RNA-seq expression data". In: *bioRxiv*, p. 068478.

Li, Heng (2011). "Tabix: fast retrieval of sequence features from generic TAB-delimited files". In: *Bioinformatics* 27.5, pp. 718–719.

Kent, W James, Ann S Zweig, G Barber, Angie S Hinrichs, and Donna Karolchik (2010). "BigWig and BigBed: enabling browsing of large distributed datasets". In: *Bioinformatics* 26.17, pp. 2204–2207.

Bialecki, Andrzej, Robert Muri, and Grant Ingersoll (2012). "Apache Lucene 4". In: *Proceedings of the SIGIR 2012 Workshop on Open Source Information Retrieval*, pp. 17–24. URL: http://www.cs.otago.ac.nz/homepages/andrew/involvement/2012-SIGIR-OSIR.pdf?origin=publication_detail\

#page=22,http://lucene.apache.org/core/4_10_1/core/index.html,http://lucene.apache.org/core/4_10_1/core/org/apache/lucene/search/similarities/DefaultSimilarity.html.

Bernstein, Matthew N, AnHai Doan, and Colin N Dewey (2016). "MetaSRA: normalized sample-specific metadata for the Sequence Read Archive". In: *bioRxiv*, p. 090506.

Goldstein, Leonard D, Yi Cao, Gregoire Pau, Michael Lawrence, Thomas D Wu, Somasekar Seshagiri, and Robert Gentleman (2016b). "Prediction and Quantification of Splice Events from RNA-Seq Data". In: *PloS one* 11.5, e0156132.

SIBGenes Gene Prediction Track (2014). URL: https://genome.ucsc.edu/cgi-bin/hgTrackUi?db=hg38&g=sibGene.

Thierry-Mieg, D. and J. Thierry-Mieg (2006). "AceView: a comprehensive cDNA-supported gene and transcripts annotation". In: *Genome Biol.* 7 Suppl 1, pp. 1–14.

Darby, Miranda M, Jeffrey T Leek, Ben Langmead, Robert H Yolken, and Sarven Sabunciyan (2016). "Widespread Splicing of Repetitive Element Loci into Coding Regions of Gene Transcripts". In: *Human Molecular Genetics*, ddw321.

Wiesner, Thomas, William Lee, Anna C Obenauf, Leili Ran, Rajmohan Murali, Qi Fan Zhang, Elissa WP Wong, Wenhuo Hu, Sasinya N Scott, Ronak H Shah, et al. (2015). "Alternative transcription initiation leads to expression of a novel ALK isoform in cancer". In: *Nature* 526.7573, pp. 453–457.

Bray, Nicolas L, Harold Pimentel, Páll Melsted, and Lior Pachter (2016c). "Near-optimal probabilistic RNA-seq quantification". In: *Nature biotechnology* 34.5, pp. 525–527.

Patro, Rob, Stephen M Mount, and Carl Kingsford (2014). "Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms". In: *Nature biotechnology* 32.5, pp. 462–464.

Venables, J. P., R. Klinck, A. Bramard, L. Inkel, G. Dufresne-Martin, C. Koh, J. Gervais-Bird, E. Lapointe, U. Froehlich, M. Durand, D. Gendron, J. P. Brosseau, P. Thibault, J. F. Lucier, K. Tremblay, P. Prinos, R. J. Wellinger, B. Chabot, C. Rancourt, and S. A. Elela (2008). "Identification of alternative splicing markers for breast cancer". In: *Cancer Res.* 68.22, pp. 9525–9531.

Tyner, Cath, Galt P Barber, Jonathan Casper, Hiram Clawson, Mark Diekhans, Christopher Eisenhart, Clayton M Fischer, David Gibson, Jairo Navarro Gonzalez, Luvina Guruvadoo, et al. (2016). "The UCSC Genome Browser database: 2017 update". In: *Nucleic Acids Research*, gkw1134.

Ling, J. P., C. Wilks, R. Charles, P. J. Leavey, D. Ghosh, L. Jiang, C. P. Santiago, B. Pang, A. Venkataraman, B. S. Clark, A. Nellore, B. Langmead, and S. Blackshaw (2020). "ASCOT identifies key regulators of neuronal subtype-specific splicing". In: *Nat Commun* 11.1, p. 137.

Sloan, C. A., E. T. Chan, J. M. Davidson, V. S. Malladi, J. S. Strattan, B. C. Hitz, I. Gabdank, A. K. Narayanan, M. Ho, B. T. Lee, L. D. Rowe, T. R. Dreszer, G. Roe, N. R. Podduturi, F. Tanaka, E. L. Hong, and J. M. Cherry (2016). "ENCODE data at the ENCODE portal". In: *Nucleic Acids Res.* 44.D1, pp. D726–732.

Sundararaman, B., L. Zhan, S. M. Blue, R. Stanton, K. Elkins, S. Olson, X. Wei, E. L. Van Nostrand, G. A. Pratt, S. C. Huelga, B. M. Smalec, X. Wang, E. L. Hong, J. M. Davidson, E. L?cuyer, B. R. Graveley, and G. W. Yeo (2016). "Resources for the Comprehensive Discovery of Functional RNA Elements". In: *Mol. Cell* 61.6, pp. 903–913.

Madugundu, A. K., C. H. Na, R. S. Nirujogi, S. Renuse, K. P. Kim, K. H. Burns, C. Wilks, B. Langmead, S. E. Ellis, L. Collado-Torres, M. K. Halushka, M. S. Kim, and A. Pandey (2019). "Integrated Transcriptomic and Proteomic Analysis of Primary Human Umbilical Vein Endothelial Cells". In: *Proteomics*, e1800315.

Huber, W., V. J. Carey, R. Gentleman, S. Anders, M. Carlson, B. S. Carvalho, H. C. Bravo, S. Davis, L. Gatto, T. Girke, R. Gottardo, F. Hahne, K. D. Hansen, R. A. Irizarry, M. Lawrence, M. I. Love, J. MacDonald, V. Obenchain, A. K. Ole?, H. Pag?s, A. Reyes, P. Shannon, G. K. Smyth, D. Tenenbaum, L. Waldron, and M. Morgan (2015). "Orchestrating high-throughput genomic analysis with Bioconductor". In: *Nat. Methods* 12.2, pp. 115–121.

Wilks, C., P. Gaddipati, A. Nellore, and B. Langmead (2018). "Snaptron: querying splicing patterns across tens of thousands of RNA-seq samples". In: *Bioinformatics* 34.1, pp. 114–116.

Burke, E. E., J. G. Chenoweth, J. H. Shin, L. Collado-Torres, S. K. Kim, N. Micali, Y. Wang, C. Colantuoni, R. E. Straub, D. J. Hoeppner, H. Y. Chen, A. Sellers, K. Shibbani, G. R. Hamersky, M. Diaz Bustamante, B. N. Phan, W. S. Ulrich, C. Valencia, A. Jaishankar, A. J. Price, A. Rajpurohit, S. A. Semick, R. W. B?rli, J. C. Barrow, D. J. Hiler, S. C. Page, K. Martinowich, T. M. Hyde, J. E. Kleinman, K. F. Berman, J. A. Apud, A. J. Cross, N. J. Brandon, D. R. Weinberger, B. J. Maher, R. D. G. McKay, and A. E. Jaffe (2020). "Dissecting transcriptomic signatures of neuronal differentiation and maturation using iPSCs". In: *Nat Commun* 11.1, p. 462.

Köster, J. and S. Rahmann (2018). "Snakemake-a scalable bioinformatics workflow engine". In: *Bioinformatics* 34.20, p. 3600.

Afgan, E., D. Baker, B. Batut, M. van den Beek, D. Bouvier, M. Cech, J. Chilton, D. Clements, N. Coraor, B. A. Gr?ning, A. Guerler, J. Hillman-Jackson, S. Hiltemann, V. Jalili, H. Rasche, N. Soranzo, J. Goecks, J. Taylor, A. Nekrutenko, and D. Blankenberg (2018). "The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update". In: *Nucleic Acids Res.* 46.W1, W537–W544.

The GTEx Consortium (2013). "The Genotype-Tissue Expression (GTEx) project". In: *Nat. Genet.* 45.6, pp. 580–585.

Network, Cancer Genome Atlas Research, John N Weinstein, Eric A Collisson, Gordon B Mills, Kenna R Mills Shaw, Brad A Ozenberger, Kyle Ellrott, Ilya Shmulevich, Chris Sander, and Joshua M Stuart (2013). "The Cancer Genome Atlas Pan-Cancer analysis project." English. In: *Nature Genetics* 45.10, pp. 1113–1120. DOI: 10.1038/ng.2764. URL: http://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=pubmed&id=24071849&retmode=ref&cmd=prlinks.

Lachmann, A., D. Torre, A. B. Keenan, K. M. Jagodnik, H. J. Lee, L. Wang, M. C. Silverstein, and A. Ma'ayan (2018). "Massive mining of publicly available RNA-seq data from human and mouse". In: *Nat Commun* 9.1, p. 1366.

Lachmann, Alexander, Zhuorui Xie, and Avi Ma'ayan (2018). "Elysium: RNA-seq Alignment in the Cloud". In: *bioRxiv*, p. 382937.

Bray, N. L., H. Pimentel, P. Melsted, and L. Pachter (2016b). "Near-optimal probabilistic RNA-seq quantification". In: *Nat. Biotechnol.* 34.5, pp. 525–527.

Dobin, A. and T. R. Gingeras (2016). "Optimizing RNA-Seq Mapping with STAR". In: *Methods Mol. Biol.* 1415, pp. 245–262.

Tatlow, P. J. and S. R. Piccolo (2016). "A cloud-based workflow to quantify transcript-expression levels in public cancer compendia". In: *Sci Rep* 6, p. 39259.

Barretina, J., G. Caponigro, N. Stransky, K. Venkatesan, A. A. Margolin, S. Kim, C. J. Wilson, J. Lehar, G. V. Kryukov, D. Sonkin, A. Reddy, M. Liu, L. Murray, M. F. Berger, J. E. Monahan, P. Morais, J. Meltzer, A. Korejwa, J. Jane-Valbuena, F. A. Mapa, J. Thibault, E. Bric-Furlong, P. Raman, A. Shipway, I. H. Engels, J. Cheng, G. K. Yu, J. Yu, P. Aspesi, M. de Silva, K. Jagtap, M. D. Jones, L. Wang, C. Hatton, E. Palescandolo, S. Gupta, S. Mahan, C. Sougnez, R. C. Onofrio, T. Liefeld, L. MacConaill, W. Winckler, M. Reich, N. Li, J. P. Mesirov, S. B. Gabriel, G. Getz, K. Ardlie, V. Chan, V. E. Myer, B. L. Weber, J. Porter, M. Warmuth, P. Finan, J. L. Harris, M.

Meyerson, T. R. Golub, M. P. Morrissey, W. R. Sellers, R. Schlegel, and L. A. Garraway (2012). "The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity". In: *Nature* 483.7391, pp. 603–607.

Vivian, J., A. A. Rao, F. A. Nothaft, C. Ketchum, J. Armstrong, A. Novak, J. Pfeil, J. Narkizian, A. D. Deran, A. Musselman-Brown, H. Schmidt, P. Amstutz, B. Craft, M. Goldman, K. Rosenbloom, M. Cline, B. O'Connor, M. Hanna, C. Birger, W. J. Kent, D. A. Patterson, A. D. Joseph, J. Zhu, S. Zaranek, G. Getz, D. Haussler, and B. Paten (2017). "Toil enables reproducible, open source, big biomedical data analyses". In: *Nat. Biotechnol.* 35.4, pp. 314–316.

Li, B. and C. N. Dewey (2011). "RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome". In: *BMC Bioinformatics* 12, p. 323.

Petryszak, R., N. A. Fonseca, A. Füllgrabe, L. Huerta, M. Keays, Y. A. Tang, and A. Brazma (2017). "The RNASeq-er API-a gateway to systematically updated analysis of public RNA-seq data". In: *Bioinformatics* 33.14, pp. 2218–2220.

Papatheodorou, I., P. Moreno, J. Manning, A. M. Fuentes, N. George, S. Fexova, N. A. Fonseca, A. F?llgrabe, M. Green, N. Huang, L. Huerta, H. Iqbal, M. Jianu, S. Mohammed, L. Zhao, A. F. Jarnuczak, S. Jupp, J. Marioni, K. Meyer, R. Petryszak, C. A. Prada Medina, C. Talavera-L?pez, S. Teichmann, J. A. Vizcaino, and A. Brazma (2020). "Expression Atlas update: from tissues to single cells". In: *Nucleic Acids Res.* 48.D1, pp. D77–D83.

Barrett, T., S. E. Wilhite, P. Ledoux, C. Evangelista, I. F. Kim, M. Tomashevsky, K. A. Marshall, K. H. Phillippy, P. M. Sherman, M. Holko, A. Yefanov, H. Lee, N. Zhang, C. L. Robertson, N. Serova, S. Davis, and A. Soboleva (2013). "NCBI GEO: archive for functional genomics data sets–update". In: *Nucleic Acids Res.* 41.Database issue, pp. D991–995.

Athar, A., A. Fullgrabe, N. George, H. Iqbal, L. Huerta, A. Ali, C. Snow, N. A. Fonseca, R. Petryszak, I. Papatheodorou, U. Sarkans, and A. Brazma (2019). "ArrayExpress update - from bulk to single-cell expression data". In: *Nucleic Acids Res.* 47.D1, pp. D711–D715.

Srivastava, A., L. Malik, T. Smith, I. Sudbery, and R. Patro (2019). "Alevin efficiently estimates accurate gene abundances from dscRNA-seq data". In: *Genome Biol.* 20.1, p. 65.

Goetz, J.J. and J.M. Trimarchi (2012). "Transcriptome sequencing of single cells with Smart-Seq". In: *Nature biotechnology* 30.8, pp. 763–765.

Picelli, S., Å.K. Björklund, O.R. Faridani, S. Sagasser, G. Winberg, and R. Sandberg (2013). "Smart-seq2 for sensitive full-length transcriptome profiling in single cells". In: *Nature methods* 10.11, pp. 1096–1098.

O'Leary, N. A., M. W. Wright, J. R. Brister, S. Ciufo, D. Haddad, R. McVeigh, B. Rajput, B. Robbertse, B. Smith-White, D. Ako-Adjei, A. Astashyn, A. Badretdin, Y. Bao, O. Blinkova, V. Brover, V. Chetvernin, J. Choi, E. Cox, O. Ermolaeva, C. M. Farrell, T. Goldfarb, T. Gupta, D. Haft, E. Hatcher, W. Hlavina, V. S. Joardar, V. K. Kodali, W. Li, D. Maglott, P. Masterson, K. M. McGarvey, M. R. Murphy, K. O'Neill, S. Pujar, S. H. Rangwala, D. Rausch, L. D. Riddick, C. Schoch, A. Shkeda, S. S. Storz, H. Sun, F. Thibaud-Nissen, I. Tolstoy, R. E. Tully, A. R. Vatsan, C. Wallin, D. Webb, W. Wu, M. J. Landrum, A. Kimchi, T. Tatusova, M. DiCuccio, P. Kitts, T. D. Murphy, and K. D. Pruitt (2016). "Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation". In: *Nucleic Acids Res.* 44.D1, pp. D733–745.

Frankish, A., M. Diekhans, A. M. Ferreira, R. Johnson, I. Jungreis, J. Loveland, J. M. Mudge, C. Sisu, J. Wright, J. Armstrong, I. Barnes, A. Berry, A. Bignell, S. Carbonell Sala, J. Chrast, F. Cunningham, T. Di Domenico, S. Donaldson, I. T. Fiddes, C. Garc?a Gir?n, J. M. Gonzalez, T. Grego, M. Hardy, T. Hourlier, T. Hunt, O. G. Izuogu, J. Lagarde, F. J. Martin, L. Mart?nez, S. Mohanan, P. Muir, F. C. P. Navarro, A. Parker, B. Pei, F. Pozo, M. Ruffier, B. M. Schmitt, E. Stapleton, M. M. Suner, I. Sycheva, B. Uszczynska-Ratajczak, J. Xu, A. Yates, D. Zerbino, Y. Zhang, B. Aken, J. S. Choudhary, M. Gerstein, R. Guig?, T. J. P. Hubbard, M. Kellis, B. Paten, A. Reymond, M. L. Tress, and P. Flicek (2019). "GENCODE reference annotation for the human and mouse genomes". In: *Nucleic Acids Res.* 47.D1, pp. D766–D773.

Hon, C., J.A. Ramilowski, J. Harshbarger, N. Bertin, O.J.L. Rackham, J. Gough, E. Denisenko, S. Schmeier, T.M. Poulsen, J. Severin, et al. (2017). "An atlas of human long non-coding RNAs with accurate 5′ ends". In: *Nature* 543.7644, pp. 199–204.

Collado-Torres, L., A. Nellore, A.C. Frazee, C. Wilks, M.I. Love, B. Langmead, R.A. Irizarry, J.T. Leek, and A.E. Jaffe (2017a). "Flexible expressed region analysis for RNA-seq with derfinder". In: *Nucleic acids research* 45.2, e9–e9.

Ellis, S. E., L. Collado-Torres, A. Jaffe, and J. T. Leek (2018a). "Improving the value of public RNA-seq expression data by phenotype prediction". In: *Nucleic Acids Res.* 46.9, e54.

Baker, D. N. and B. Langmead (2019). "Dashing: fast and accurate genomic distances with HyperLogLog". In: *Genome Biol.* 20.1, p. 265.

Ondov, B. D., T. J. Treangen, P. Melsted, A. B. Mallonee, N. H. Bergman, S. Koren, and A. M. Phillippy (2016). "Mash: fast genome and metagenome distance estimation using MinHash". In: *Genome Biol.* 17.1, p. 132.

Li, H. (2020 (accessed August 18, 2020)). *seqtk: Toolkit for processing sequences in FASTA/Q formats*. URL: https://github.com/lh3/seqtk.

Ziemann, M., A. Kaspi, and A. El-Osta (2019). "Digital expression explorer 2: a repository of uniformly processed RNA sequencing data". In: *Gigascience* 8.4.

Mantere, T., S. Kersten, and A. Hoischen (2019). "Long-Read Sequencing Emerging in Medical Genetics". In: *Front Genet* 10, p. 426.

Amarasinghe, S. L., S. Su, X. Dong, L. Zappia, M. E. Ritchie, and Q. Gouil (2020). "Opportunities and challenges in long-read sequencing data analysis". In: *Genome Biol.* 21.1, p. 30.

Kovaka, S., A. V. Zimin, G. M. Pertea, R. Razaghi, S. L. Salzberg, and M. Pertea (2019). "Transcriptome assembly from long-read RNA-seq alignments with StringTie2". In: *Genome Biol.* 20.1, p. 278.

Langmead, B. (2017). "A tandem simulation framework for predicting mapping quality". In: *Genome Biol.* 18.1, p. 152.

Langmead, B. and S. L. Salzberg (2012). "Fast gapped-read alignment with Bowtie 2". In: *Nat. Methods* 9.4, pp. 357–359.

Li, Heng (2013). "Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM". In: *arXiv preprint arXiv:1303.3997*.

Zaharia, Matei, William J Bolosky, Kristal Curtis, Armando Fox, David Patterson, Scott Shenker, Ion Stoica, Richard M Karp, and Taylor Sittler (2011). "Faster and more accurate sequence alignment with SNAP". In: *arXiv preprint arXiv:1111.5572*.

Tang, A. D., C. M. Soulette, M. J. van Baren, K. Hart, E. Hrabeta-Robinson, C. J. Wu, and A. N. Brooks (2020). "Full-length transcript characterization of SF3B1 mutation in chronic lymphocytic leukemia reveals downregulation of retained introns". In: *Nat Commun* 11.1, p. 1438.

Tardaguila, M., L. de la Fuente, C. Marti, C. Pereira, F. J. Pardo-Palacios, H. Del Risco, M. Ferrell, M. Mellado, M. Macchietto, K. Verheggen, M. Edelmann, I. Ezkurdia, J. Vazquez, M. Tress, A. Mortazavi, L. Martens, S. Rodriguez-Navarro, V. Moreno-Manzano, and A. Conesa (2018). "SQANTI: extensive characterization of long-read transcript sequences for quality control in full-length transcriptome identification and quantification". In: *Genome Res.*

Darby, C. A., R. Gaddipati, M. C. Schatz, and B. Langmead (2020). "Vargas: heuristic-free alignment for assessing linear and graph read aligners". In: *Bioinformatics* 36.12, pp. 3712–3718.

Smith, T. F. and M. S. Waterman (1981). "Identification of common molecular subsequences". In: *J. Mol. Biol.* 147.1, pp. 195–197.

Jeffares, D. C., C. Jolly, M. Hoti, D. Speed, L. Shaw, C. Rallis, F. Balloux, C. Dessimoz, J. B?hler, and F. J. Sedlazeck (2017b). "Transient structural variations have strong effects on quantitative traits and reproductive isolation in fission yeast". In: *Nat Commun* 8, p. 14061.

Gordon, S. P., E. Tseng, A. Salamov, J. Zhang, X. Meng, Z. Zhao, D. Kang, J. Underwood, I. V. Grigoriev, M. Figueroa, J. S. Schilling, F. Chen, and Z. Wang (2015). "Widespread Polycistronic Transcripts in Fungi Revealed by Single-Molecule mRNA Sequencing". In: *PLoS ONE* 10.7, e0132628.

Workman, R. E., A. D. Tang, P. S. Tang, M. Jain, J. R. Tyson, R. Razaghi, P. C. Zuzarte, T. Gilpatrick, A. Payne, J. Quick, N. Sadowski, N. Holmes, J. G. de Jesus, K. L. Jones, C. M. Soulette, T. P. Snutch, N. Loman, B. Paten, M. Loose, J. T. Simpson, H. E. Olsen, A. N. Brooks, M. Akeson, and W. Timp (2019). "Nanopore native RNA sequencing of a human poly(A) transcriptome". In: *Nat. Methods* 16.12, pp. 1297–1305.

Roach, N. P., N. Sadowski, A. F. Alessi, W. Timp, J. Taylor, and J. K. Kim (2020). "The full-length transcriptome of C. elegans using direct RNA sequencing". In: *Genome Res.* 30.2, pp. 299–312.

Zheng, G. X., J. M. Terry, P. Belgrader, P. Ryvkin, Z. W. Bent, R. Wilson, S. B. Ziraldo, T. D. Wheeler, G. P. McDermott, J. Zhu, M. T. Gregory, J. Shuga, L. Montesclaros, J. G. Underwood, D. A. Masquelier, S. Y. Nishimura, M. Schnall-Levin, P. W. Wyatt, C. M. Hindson, R. Bharadwaj, A. Wong, K. D. Ness, L. W. Beppu, H. J. Deeg, C. McFarland, K. R. Loeb, W. J. Valente, N. G. Ericson, E. A. Stevens, J. P. Radich, T. S. Mikkelsen, B. J. Hindson, and J. H. Bielas (2017). "Massively parallel digital transcriptional profiling of single cells". In: *Nat Commun* 8, p. 14049.

Ellis, S. E., L. Collado-Torres, A. Jaffe, and J. T. Leek (2018b). "Improving the value of public RNA-seq expression data by phenotype prediction". In: *Nucleic Acids Res.* 46.9, e54.

Appendices

Appendix A

Additional Details of the Monorail Ecosystem

A.1 Selection of SRA datasets

For the SRA human and mouse compilations, we downloaded and filtered a set of sequencing runs from the SRA summarized in Table A.1 and visualized in Figure A.1.

Table A.1: SRA Metadata Queried & Processed

	Runs	Studies	TeraBases	TeraBytes (compressed)
Pre-scRNA filtered Human	493374	9401	1082	569
Pre-scRNA filtered Mouse	676653	11512	935	465
Filtered Bulk Human	218982	8357	868	451
Filtered smartSeq Human	97467	320	60	30
Filtered Bulk Mouse	203170	9407	657	329
Filtered smartSeq Mouse	213733	681	75	34

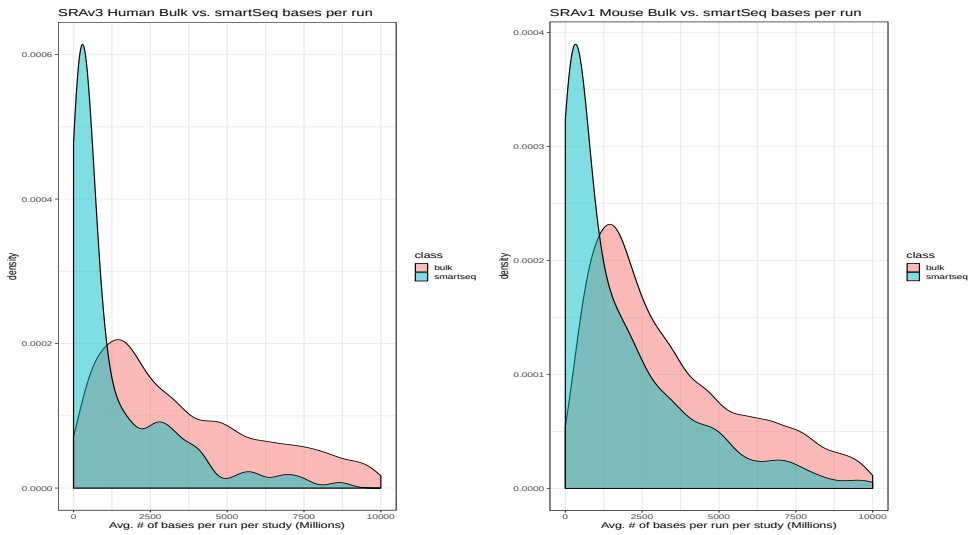

Figure A.1: Human & Mouse Average per run density across studies

A.2 Obtaining GTEx and TCGA data & metadata

We obtained GTEx metadata from the "Annotations" section of the GTEx portal: https://storage.googleapis.com/gtex_analysis_v8/annotations/GTEx_Analysis_v8_Annotations_SampleAttributesDS.txt

Since GTEx includes multiple runs per aliquot, we extended the GTEx aliquot barcode with a ".#" to indicate which run the barcode is referring to. This is called "rail_barcode" in our files.

At the time of data collections, samples from all GTEx releases up to and including V7 were accessioned by the SRA and visible in the SRA Run Browser. Samples in GTExV8 (excluding V6 & V7) were not accessioned in the SRA, and are not present on the SRA Run Browser. Sequence data for GTExV7 & V8 samples (excluding V6) were available only in the AWS (V7 only) or Google Cloud Platform (GCP, V7 & V8) commercial clouds. We retrieved GTEx V7 and V8 sequence data from GCP as BAM files (9,303 files), and retrieved all the sample sequence data up to and including V6 from the SRA directly in the normal format (9,911 files). We used the "gsutil cp" tool to download from GCP, and the "prefetch" tool from the SRA-Toolkit (together with "parallel-fastq-dump") to obtain FASTQ data from the SRA. The SRA retrieval tools are part of the Monorail Docker image; we do not include "gsutil" as we

considered its use to be a one-time event.

Metadata for TCGA was inherited directly from recount2 (Collado-Torres et al., 2017). TCGA sample sequence data was downloaded from the Genome Data Commons (GDC) using the GDC Download Client tool, version 1.4, also included in the Monorail Docker image.

A.3 Quality control

We used a number of tools to collect potentially useful quality-control measures. Specifically, we used `seqtk` (Li, 2020 (accessed August 18, 2020)), the `idxstats` subcommand of `samtools`, the output of `STAR`, our own `megadepth` tool, and `featureCounts`. We examine each in turn, listing the specific QC measures calculated be each.

Monorail runs the `seqtk fqchk` command on input FASTQ files to collect base-quality and base-composition summaries for all sequencing cycles. We distill these into a few QC measures included with every summarized run in recount3.

Monorail uses `STAR` to align RNA-seq reads in a spliced fashion to a reference genome, without using any annotation. Files output by `STAR`, particularly the `Log.out` and `Log.final.out`, report a number of measures that can be used for QC. We compile these into a number of QC measures included with every summarized run in recount3.

From the STAR manual (version 2.7.2b): "Log.final.out: summary mapping statistics after mapping job is complete, very useful for quality control. The statistics are calculated for each read (single- or paired-end) and then summed or averaged over all reads. Note that STAR counts a paired-end read as one read, (unlike the samtools flagstat/idxstats, which count each mate separately). Most of the information is collected about the UNIQUE mappers (unlike samtools flagstat/idxstats which does not separate unique or multi-mappers). Each splicing is counted in the numbers of splices, which would correspond to summing the counts in SJ.out.tab. The mismatch/indel error rates are calculated on a per base basis, i.e. as total number of mismatches/indels in all unique mappers divided by the total number of mapped bases." Some of the following definitions include text from the STAR manual/source code, reprinted here for convenience. Please see the STAR manual for more in depth information.

Monorail runs the `samtools idxstats` on the BAM file output by `STAR` to collect statistics about how many reads aligned to each chromosome in the genome assembly. This can be helpful in, for instance, confirming the sex of the individual sequenced based on alignments to sex chromosomes, or measuring effectiveness of ribosomal RNA depletion by considering the fraction of reads aligned to the mitochondrial genome. We compile these into a number of QC measures included with every summarized run in recount3.

Monorail runs our `megadepth` tool on the BAM files output by `STAR`. The chief function is to convert BAM files to bigWig files that are then added to the recount3 archive. As `megadepth` performs this conversion, it also summarizes the amount of sequencing coverage within the intervals of a provided BED file representing a gene annotation. These quantifications can be useful for quality control, tell us, for example, what fraction of the coverage is within annotated genes.

Fragment length distribution is based on a special read filter only applied for this purpose to be compatible with CSAW's fragment counting approach (Lun and Smyth, 2016), paired reads in a passing fragment must not be secondary, supplementary, have conflicting read order, be unmapped or be mapped on more than one chromosome.

Finally, Monorail runs `featureCounts` on the BAM files output by `STAR`. This provides a "second opinion" on the quantifications produced by `megadepth`. While we have not yet found compelling examples where the `megadepth` and `featureCounts` outputs disagree, we keep summaries of the `featureCounts` quantifications as potential QC measures.

A.4 Monorail workflow specifics

Here we describe the design and implementation of Monorail in detail. We focus on portions of the system that are relevant to the outputs needed for recount3 and Snaptron. The system has additional tools and features that are not described here, but these are experimental and/or not required to produce the standard RNA-seq summaries needed for recount3.

A.4.1 Orchestration

Monorail follows a grid computing design, meaning that computational tasks can take place on various systems at various times, with all computation coordinated over the Internet by a few centralized services. Monorail's centralized components run on Amazon Web Services. A **database server** hosts a database containing the overall data model, discussed later. This is a `db.t2.medium` instance from the Amazon Relational Database Service (RDS) running PostgreSQL version 10. A **job queue** provides a centralized, synchronized way for various analysis nodes to obtain the next available unit of work. Since it is synchronized, there is no chance of a "race condition" in the event that many analysis nodes ask for the next unit of work at the same time. This facility uses AWS's Simple Queue Service (SQS), which also provides a degree of fault tolerance via timeout and job-visibility mechanisms. A **reference file repository** stores the reference files — e.g. genome assembly FASTA files, index files, gene annotation files — used across the project. This uses AWS's Simple Storage Service (S3). Finally, a **centralized logging service** provides a single place for all the components of the system to keep logs. Analysis nodes, orchestration services, and client software all archive messages in this central repository. We use the AWS CloudWatch service for this facility. CloudWatch additionally allows us to visually follow the state of the system by viewing a CloudWatch Dashboard. This is pictured in Figure 3.9.

A.4.2 Data Model

The Monorail data model, pictured in Figure A.2, defines the kinds of information can be tracked by the orchestration layer. For instance, the **`input` table** describes all the sequencing-read input files for all the computations. For some files, these might "point to" the dataset via an SRA accession; for others, this might use a URL to locate the file. The **`annotation` and `source` tables** contain information about the origin of all the reference files used, including genome indexes and gene annotations. The **`analysis`** table describes all the Docker and/or Singularity images that might be used to analyze an input dataset. The data model can be created and modified using Python scripts in the orchestration software, available at `https://github.com/langmead-lab/recount-pump`. This software uses the SQLAlchemy object-relational model to map tables in the PostgreSQL database to objects in the Python infrastructure.

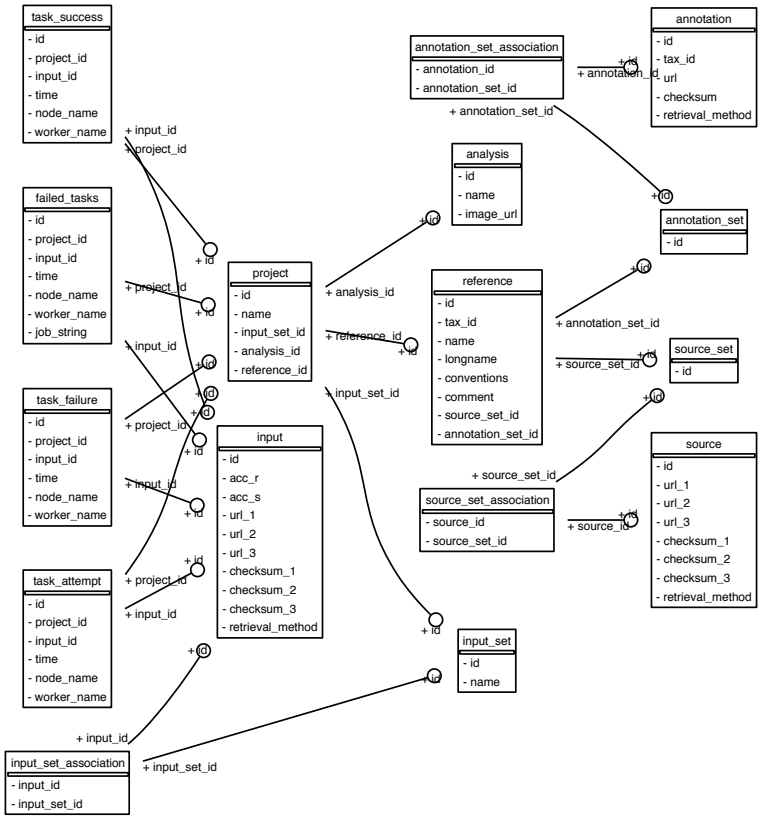

Figure A.2: The Monorail relational database model. Rectangles denote tables and arcs denote the key relationships between tables. Image was created using the `sqlalchemy_schemadisplay` package.

A.4.3 Managers and runners

Now we describe the software that runs on the compute clusters that obtain jobs from the orchestration layer, perform the analysis, and store the outputs. The the highest level, an analysis node runs a **node manager**, which launches a number of individual "job runners," each allocated a fraction of available memory and hardware threads. The bottom layer is the **runner** which runs on the compute-node "slice" allocated to it by the node manager. A runner enters a "job loop," where it repeatedly checks a queue of all pending tasks for the project. A task is a single dataset to be analyzed. This design is illustrated at two levels of granularity in Figures 3.5 and 3.7. Once it has obtained a job, a runner launches a Singularity container that in turn runs the corresponding workflow.

A.4.4 Workflow

The Monorail analysis workflow is driven by a Snakemake workflow that runs inside a Singularity container. The use of a container system allows us to package all of the constituent software tools and all their dependencies in a single image. We use singularity in particular (rather than Docker, for example) because Singularity is designed to be able to run with non-root privileges on multi-user cluster computing systems. We find that Singularity is commonly available on scientific clusters including our local MARCC cluster and the XSEDE supercomputers.

A.4.5 Aggregation

The runners produce output files that are either transferred immediately to the aggregation node via Globus, or stored locally in preparation for a periodic bulk transfer. At this point, we run the aggregator software in order to combine the output files into the tables and indexes required for recount3/Snaptron.

We initially started with every file being transferred by Globus to the aggregation filesystem after a specific sequence run job had finished. However, this put undue load on the Globus API service which allowed for a limited number of pending transfer requests and concurrent API connections, thus causing workflow failures for specific jobs. We then switched to keeping all finished jobs on storage local to the compute environment run on (e.g. scratch storage in TACC Stampede2). Then after a full tranche of jobs was finished,

we'd batch transfer the whole output via a single Globus job, this worked mostly without issue.

Once all output files for a specific batch were Globus transferred to the aggregation filesystem, we started a run of the "recount-unifier" which did the following steps (also illustrated in Figure 3.8):

- Decompress Exon & Junction coverage files
- Paste exon sums together
- Rejoin disjoint exon sums into originally annotated gene and exon sum matrices, split by study
- Merge junctions and their split read counts into a sparse matrix only including samples which had > 0 splits reads for a junction
- Add junction annotation abbreviations for Snaptron
- Split junction coverage into per-study sparse matrices in the Matrix Market format for recount

The same output from the recount-unifier feeds both the Snaptron and recount projects, though formatted differently. Further, QC statistics are aggregated across sequence runs for the tranche. In addition to Snakemake, we also use the GNU parallel utility (Tange et al., 2011) heavily in the recount-unifier part of the workflow.

A.5 Genome Reference Annotation Files

The human gene annotations were chosen to represent a reasonably extensive representation of the state of the field at the time (February 2019).

The following GRCh38 (HG38) annotations were used:

- Gencode V26 (G026)
- Gencode V29 (G029)
- RefSeq (R109)
- FANTOM-CAT V6 (F006)

We included Gencode V26 as our main annotation reference due to its use in the GTExV8 project. The link downloaded from was ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_26/gencode.v26.chr_patch_hapl_scaff.annotation.gtf.gz.

Gencode V29 and RefSeq 109 were chosen to be recent versions that matched with the genome assembly we used. We also included FANTOM-CAT (v6) (Hon et al., 2017) as an annotation that is more inclusive of non-coding RNA. For QC & controls we included the synthetic genes from the ERCC (listed, 2005) project and the synthetic exons from the SIRV transcriptome project (Byrne et al., 2017).

We chose a single, recent Gencode version for our mouse annotation (M23) based on GRCm38 (mm10) in addition to the ERCC and SIRV sets mentioned above. We also checked the genomic sequence between GRCm38 and mm10 for any differences for the chromosomes and contigs we used in alignment and there were none, so we consider them equivalent for the purposes of this project. The link used to download M23 was: ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_mouse/release_M23/gencode.vM23.primary_assembly.annotation.gtf.gz.

For both organisms, we converted a GTF of the combined set of annotations above to the GFF3 format via the makeTxDbFromGFF function in the GenomicFeatures package. We then passed that GFF3-formatted data to the exonicParts function also from the GenomicFeatures package setting "linked.to.single.gene.only" to FALSE. This produced a set of disjoint exons across genes across all the annotations.

Sums were generated over this disjoint set of exons via the Megapdeth tool for every BAM temporary produced in the Monorail workflow. These exon sums were then pasted together across all samples in a dataset (e.g. SRAv3) and then "rejoined" into their original annotated exon and gene sums. These latter steps occur in the aggregation portion of the Monorail workflow.

For junctions, we expand the set of annotations we use for both human and mouse to more broadly cover as much annotated splicing as possible. The approach that produced the following list (Table A.2) was based directly on (Nellore et al., 2016) which in turn was influenced by (Farkas et al., 2013).

A.6 BigWig processing with Megadepth

Megadepth is a custom tool we built primarily to serve two main purposes in this project:

- Efficiently extract coverage summaries from the temporary BAM files produced in Monorail and write them out as BigWigs
- Efficiently re-quantify coverage over the BigWig files produced in the previous step for a new annotation/set of intervals, avoiding the re-downloading of the original sequence + alignment steps

While there are tools available to perform both of these functions (e.g. pyBigWig, wiggletools), no one tool does them all. Additionally, no other tool, to our knowledge, does direct conversion of coverage in a BAM file into a BigWig file, Megadepth does this among other functions.

For the first purpose above, using off the shelf tools would've required at least 2 separate applications—and likely more—to accomplish what is being done in Megadepth. Additionally, not only would the dependencies of the Monorail workflow be more complex without Megadepth, it would be less efficient, because each tool would need to make a separate pass through the BAM file, for every sample process through Monorail.

Table A.2: Junction annotation sources. Descriptions are from the UCSC Table Browser track detail page or the Gencode website

Short Name	Description	Reference Build
Acembly	AceView gene models constructed from cDNA by Danielle and Jean Thierry-Mieg at NCBI, using their AceView program	hg19, mm9
Chess 2.2	Chess transcripts assembled using StringTie based on GTEx (Pertea et al., 2018)	hg38
ccdsGene	Human genome high-confidence gene annotations from the Consensus Coding Sequence (CCDS) project	hg19, hg38, mm9, mm10
Gencode	19 (hg19), 24-26, 29, 33 (hg38) 1 (mm9), 2-24 (mm10)	hg19, hg38, mm9, mm10
GSE72311_lncrna	long non-coding RNA transcripts from the GSE72311 study	mm10
knownGene	A set of UCSC gene predictions based on data from RefSeq, GenBank, CCDS, Rfam, and the tRNA Genes track	hg19, hg38, mm9, mm10
lincRNAsTranscripts	Human Body Map lincRNAs (large intergenic non coding RNAs) and TUCPs (transcripts of uncertain coding potential)	hg19, hg38
mgcGenes	The Mammalian Gene Collection (MGC) of full-length open reading frames (ORFs) in the genome.	hg19, hg38, mm9, mm10
refGene	The NCBI RNA reference sequences collection (RefSeq)	hg19, hg38, mm9, mm10
sibGene	Swiss Institute of Bioinformatics cDNA/EST-based gene predictions	hg19, hg38
vegaGene	Annotated genes from the Vertebrate Genome Annotation (VEGA) database (Human chr14, 20, 22 only)	hg19, mm9

Table A.3: Supplemental Table Human Annotated Junction Percentages

PercentOfSamples	Annotated	ExonSkip	OneAnnotated	NeitherAnnotated
1	54.4	10.7	24.8	10.0
2	67.7	8.7	17.7	5.9
5	84.0	5.0	8.5	2.5
10	93.4	2.4	3.2	1.0
20	98.3	0.8	0.6	0.3

Megapdeth produces several coverage summaries from its single pass through a BAM file, but the ones used in the recount3/Snaptron2 datasets are the following:

- Area Under Coverage (AUC)—related to mapping depth and used extensively in recount2
- Per-base coverage as a BigWig
- Coverage across the disjoined exons from the annotations described elsewhere in this Supplement

For all coverage summaries listed above, Megadepth reports the number(s) for all reads mapping and those reads which mapped with minimum quality $>= 10$ separately (6 different reports).

A.7 recount3 data formatting

The coverage summaries provided in recount3 are stored as tab delimited matrices in GZip compressed flat files. Rows are genes or exons, and columns are samples. Coverage is stored as raw per-base counts summed over the relevant annotation interval (gene or exon). Junction files follow the Market Matrix format which represents the junction coverage matrix as a sparse list of matrix coordinates for those cells which are non-0. The non-0 values represent the raw count of split reads supporting a given junction. Per-base coverage values are stored in BigWigs, one BigWig file per sample.

Table A.4: Supplemental Table Mouse Annotated Junction Percentages

PercentOfSamples	Annotated	ExonSkip	OneAnnotated	NeitherAnnotated
1	55.7	8.7	23.9	11.6
2	70.6	6.8	16.5	6.2
5	87.5	3.5	7.0	2.0
10	95.4	1.5	2.4	0.7
20	98.8	0.5	0.4	0.2

References

Collado-Torres, L., A. Nellore, K. Kammers, S. E. Ellis, M. A. Taub, K. D. Hansen, A. E. Jaffe, B. Langmead, and J. T. Leek (2017). "Reproducible RNA-seq analysis using recount2". In: *Nat. Biotechnol.* 35.4, pp. 319–321.

Li, H. (2020 (accessed August 18, 2020)). *seqtk: Toolkit for processing sequences in FASTA/Q formats*. URL: https://github.com/lh3/seqtk.

Lun, A. T. and G. K. Smyth (2016). "csaw: a Bioconductor package for differential binding analysis of ChIP-seq data using sliding windows". In: *Nucleic Acids Res.* 44.5, e45.

Tange, Ole et al. (2011). "Gnu parallel-the command-line power tool". In: *The USENIX Magazine* 36.1, pp. 42–47.

Hon, C., J.A. Ramilowski, J. Harshbarger, N. Bertin, O.J.L. Rackham, J. Gough, E. Denisenko, S. Schmeier, T.M. Poulsen, J. Severin, et al. (2017). "An atlas of human long non-coding RNAs with accurate 5′ ends". In: *Nature* 543.7644, pp. 199–204.

listed, No authors (2005). "Proposed methods for testing and selecting the ERCC external RNA controls". In: *BMC Genomics* 6, p. 150.

Byrne, A., A. E. Beaudin, H. E. Olsen, M. Jain, C. Cole, T. Palmer, R. M. DuBois, E. C. Forsberg, M. Akeson, and C. Vollmers (2017). "Nanopore long-read RNAseq reveals widespread transcriptional variation among the surface receptors of individual B cells". In: *Nat Commun* 8, p. 16027.

Nellore, A., A. E. Jaffe, J. P. Fortin, J. Alquicira-Hernandez, L. Collado-Torres, S. Wang, R. A. Phillips Iii, N. Karbhari, K. D. Hansen, B. Langmead, and J. T. Leek (2016). "Human splicing diversity and the extent of unannotated splice junctions across human RNA-seq samples on the Sequence Read Archive". In: *Genome Biology* 17.1, p. 266.

Farkas, M. H., G. R. Grant, J. A. White, M. E. Sousa, M. B. Consugar, and E. A. Pierce (2013). "Transcriptome analyses of the human retina identify unprecedented transcript diversity and 3.5 Mb of novel transcribed sequence

via significant alternative splicing and novel genes". In: *BMC Genomics* 14, p. 486.

Pertea, M., A. Shumate, G. Pertea, A. Varabyou, F.P. Breitwieser, Y. Chang, A.K. Madugundu, A. Pandey, and S.L. Salzberg (2018). "CHESS: a new human gene catalog curated from thousands of large-scale RNA sequencing experiments reveals extensive transcriptional noise". In: *Genome biology* 19.1, pp. 1–14.

Appendix B

Additional Details of the LongTron Method

B.1 Additional information for random forest features

The following is a complete list of features used in the random forest:

1. Sequence read length including softclipping
2. # of exons
3. Aggregate exon length
4. Aggregate intron length
5. Smallest aligned segment size (exon)
6. Smallest intron size
7. Mapping quality
8. # bases overlapping with RepeatMasker annotation

Table B.1: Top 5 Most Important Features by Category. (FL=full length, nFL= fragment)

Category	First	Second	Third	Fourth	Fifth
Oxford FL 4 class	Smallest aligned segment size (exon) (10%)	Minimum exon length across all transcripts at gene locus (6%)	Aggregate intron length (5%)	Count of canonical splice motifs in region on the same strand (4%)	Log of aggregate exon length (#2) (4%)
Oxford FL 2 class	Smallest aligned segment size (exon) (12%)	Minimum exon length across all transcripts at gene locus (7%)	Aggregate intron length (5%)	Count of canonical splice motifs in region on the same strand (4%)	Log of aggregate exon length (#2) (4%)
Oxford nFL 4 class	Minimum exon length across all transcripts at gene locus (10%)	Average length across all transcripts. exons at gene locus (6%)	Average ratio of non-unique k-mers per transcript base pair within transcripts which overlap target region (4%)	Maximum exon length across all transcripts at gene locus (4%)	Minimum transcript length across all transcripts at gene locus (4%)
Oxford nFL 2 class	Minimum exon length across all transcripts at gene locus (11%)	Average length across all transcripts. exons at gene locus (6%)	Maximum exon length across all transcripts at gene locus (4%)	Average ratio of non-unique k-mers per transcript base pair within transcripts which overlap target region (4%)	Minimum transcript length across all transcripts at gene locus (4%)
PacBio FL 4 class	Smallest aligned segment size (exon) (18%)	Minimum exon length across all transcripts at gene locus (9%)	Per-base average of GC content score in region (3%)	Aggregate intron length (3%)	Per-base average of overlapping exons (.transcript density.) (3%)
PacBio FL 2 class	Smallest aligned segment size (exon) (20%)	Minimum exon length across all transcripts at gene locus (9%)	Per-base average of GC content score in region (3%)	Aggregate intron length (3%)	Per-base average of overlapping exons (.transcript density.) (3%)
PacBio nFL 4 class	Minimum exon length across all transcripts at gene locus (11%)	Average length across all transcripts. exons at gene locus (5%)	Minimum transcript length across all transcripts at gene locus (5%)	Sum of all transcript lengths (transcript length=sum of exon lengths in transcript), this could be redundant across transcripts (5%)	Average ratio of non-unique k-mers per transcript base pair within transcripts which overlap target region (4%)
PacBio nFL 2 class	Minimum exon length across all transcripts at gene locus (12%)	Average length across all transcripts. exons at gene locus (6%)	Minimum transcript length across all transcripts at gene locus (5%)	Sum of all transcript lengths (transcript length=sum of exon lengths in transcript), this could be redundant across transcripts (5%)	Maximum exon length across all transcripts at gene locus (5%)

9. # bases overlapping with simple repeats
10. Count of overlapping common 150 SNPs
11. Count of overlapping transcripts/reads on the same strand, always has at least 1 (itself)
12. Per-base average of GC content score in region
13. Per-base average of Multi-track Mappability score k=24, umap in region
14. Per-base average of overlapping exons “exon density”
15. Per-base average of overlapping exons “transcript density”
16. Log of # of exons #1
17. Log of aggregate exon length #2
18. Log of aggregate intron length #3
19. Count of canonical splice motifs in region on the same strand
20. Count of overlapping segmental duplicates
21. Ratio of the region that overlaps segmental duplicates by base
22. Average of overlapping transcripts’ base pair length
23. Average # of unique k-mers within transcripts which overlap target region
24. Average # of non-unique k-mers within transcripts which overlap target region

25. Average ratio of non-unique k-mers per transcript base pair within transcripts which overlap target region
26. # of transcripts at gene locus
27. Sum of all transcript lengths, transcript length=sum of exon lengths in transcript, this could be redundant across transcripts
28. Minimum transcript length across all transcripts at gene locus
29. Maximum transcript length across all transcripts at gene locus
30. Average transcript length across all transcript at gene locus
31. Total number of exons across all transcript at gene locus
32. Minimum exon length across all transcripts at gene locus
33. Maximum exon length across all transcripts at gene locus
34. Average length across all transcripts' exons at gene locus
35. Distance to closest gene locus in base pairs, can be 0 or negative if closest locus is upstream

B.2 Details on junction matching

Novel: read junctions have no overlap even within a fuzz of any annotated junction (no overlap either, so these aren't within any annotated junction). These are removed from consideration in the error categories below.

Matching: Junction matching criteria:

- each end must be within the window of its annotated end
- at least one of the aligned jx's read IDs must match the annotated transcript ID it's overlapping
- the exon/jx idx must match within the overlapping transcript

The non-match from above are further split into categories:

Overlapping: a read junction which strictly overlaps an annotated junction (no containment for either).

Contained (read junction): read's junction is either fully within an annotated junction OR its ends extend beyond the annotated junction's ends, but not beyond fuzz distance of the annotated junction's ends.

Contained (annotated junction): read's junction ends are beyond the annotated junction's ends and both beyond fuzz distance of the annotated junction's ends.

B.3 gffcompare run details

In order to perform an accurate comparison at the isoform level we determined that we needed to modify an existing tool, gffcompare, part of the well known suite of isoform assembly and analysis tools Cufflinks. gffcompare at one time supported the notion of a "fuzz" parameter wherein intron boundaries with isoforms were allowed to be off by a certain length. This mode was disabled in more recent versions. To our knowledge no other tool does this. We re-enabled this mode specifically for this paper's work. This allowed us to apply the fuzz approach we took with individual splice junctions to the isoform level.

Specifically, we focused on comparing intron-chains between two isoforms. That is we ignored the start/end exons and restricted the comparison to only the coordinates of the introns.

In addition, we added a parameter which forces gffcompare to load its "reference" and "query" sets of isoforms in exactly the same way, applying the same deduplication approach to both. This ensured that the same pair of samples run in one order would be the same in the reverse. The sensitivity and precision numbers are derived from intron-chain comparison.

All comparisons are made via gffCompare (updated version of cuffCompare from Cufflinks) and are exact matches. Union of Annotations is made up of: Gencode V29, Gencode V26 (GTEx), RefSeq HG38 (as of early 2019), FANTOM-CAT 6 (lncRNA + Gencode).

B.4 Training simulation dataset pipeline

Each of the four datasets (Oxford FL, Oxford non-FL, PacBio FL, PacBio non-FL) were simulated by taking the error profile generated by SURVIVOR and using SURVIVOR's "simreads" command from the original Minimap2 alignments and using this with the set of transcript sequences from Gencode V28 to produce synthetic reads which were then aligned back to the genome using Minimap2. This process was run five separate times. Per-simulated run error categories are defined in the main text as A-E at the start of the Simulation section.

1. Original NA12878 Oxford/PacBio long reads aligned against Gencode V28 transcript sequences

2. SURVIVOR "scanreads" extracts technology specific error profile from alignments in 1. Using a read length cutoff of $>= 100$

3. SURVIVOR "simreads" then simulates new reads based on steps 1. & 2.

4. Synthetic long reads from step 3. are then mapped back to HG38 using Minimap2 using the same parameters across all runs

Steps 3-4 were run 5 times per technology for both full-length and fragments producing a total of 20 runs.

Any transcript that was consistently novel across all simulation runs was assigned to the novel class. Any transcript that was consistently in all of the error classes, but not consistently novel, was assigned to the recurrent-error class. Any transcript that was not in either the consistently novel or recurrent error classes was but had been in at least one or more error categories in one or more of the runs was assigned into the non-recurrent error class. Finally any transcript not previously filtered for was assigned a problem-free category.

The flow of decision for categorizing a single simulated transcript is as follows:

1. Consistently novel across all simulations runs? Novel (end)

2. Not in Novel and consistently in all 3 error categories across all simulation runs? Recurrent-error (end)

3. Not in Novel/Recurrent-error but was categorized in >= 1 error category in >= 1 simulation run? Any-error (end)

4. Must be Problem-free (end)

This logic is defined in the 'compare_matching.sh' script in the associated Github repository.

The original alignment of the Oxford NA12878 sample was using this command line:

minimap2 -ax splice -uf -k14 -t 16

./referenceFastaFiles/dna/GRCh38_full_analysis_set_plus_decoy_hla.fa

./NA12878-DirectRNA.pass.dedup.NoU.fastq

All simulations were run with the same parameters and the same reference.

B.5 Counting results of predictions on NA12878

Once models were trained for the four cases, these were used to predict the labels of the original Minimap2 alignments of the same samples. The RandomForest probabilities for each class for each alignment were then tabulated and a match was recorded if the class with the highest probability matched the training label or the probability the matching class was within 0.02 of the class with the highest probability.

B.6 NA12878 & SKBR Custom Tracks in the UCSC Genome Browser

NA12878 Oxford:

http://snaptron.cs.jhu.edu/data/longtron/oxford.NA12878-DirectRNA.bam.bed.sorted.bb

NA12878 PacBio:

http://snaptron.cs.jhu.edu/data/longtron/pacbo.SRR1163655.sorted.bam.noHLA.bed.bb

SKBR3 PacBio:

http://snaptron.cs.jhu.edu/data/longtron/SRR7346977_subreads.bam.nohla.bed.bb

B.7 Features used in the Random Forest training/prediction

Feature Files:

- Repeat Masker overlap: hg38_repeatmasker_rmsk
- Tandem Repeat Finder (TRF) overlap (subset of 1): simple_repeats_hg38
- Splice motif count: hg38_splice_motifs.all.bed.bgz
- Exon/intron statistics: gencode.v28.basic.annotation.exons.stats.bed (locus stats), gencode.v28.basic.annotation.exons.perbase.counts.bgz
- # of overlapping reads/transcripts:

  gencode.v28.basic.annotation.transcripts.perbase.counts.bgz

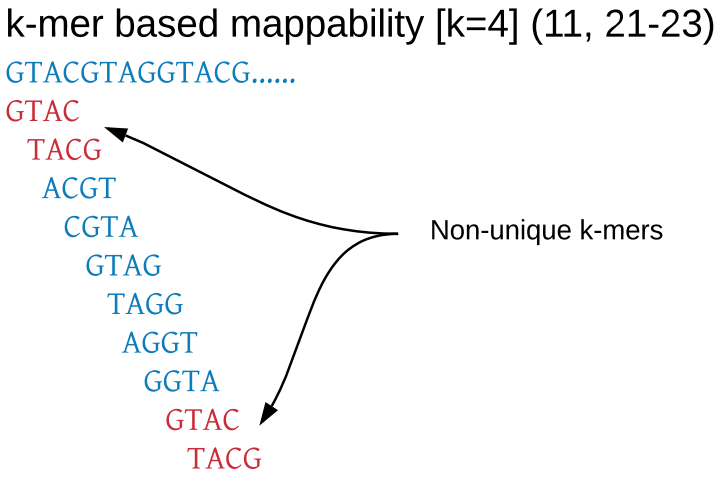

Figure B.1: Kmer mappability. Mappability is based on k-mers, k=24 for umap multi-tracking mappings and k=10 for local region mappings. This is for features used in the random forest: 11, and 21-23.

- # of common 150 SNPs overlapping:

  snp150Common.combined.sorted.bed.no_bad_chrs

- GC content: gc5Base.bg.clean

- General Mappability: k24.Umap.MultiTrackMappability.sorted.bg

- Local Mappability: gv28.local_mappability.coords.bed

- Segmental Duplications: segmental_dups_hg38.sorted

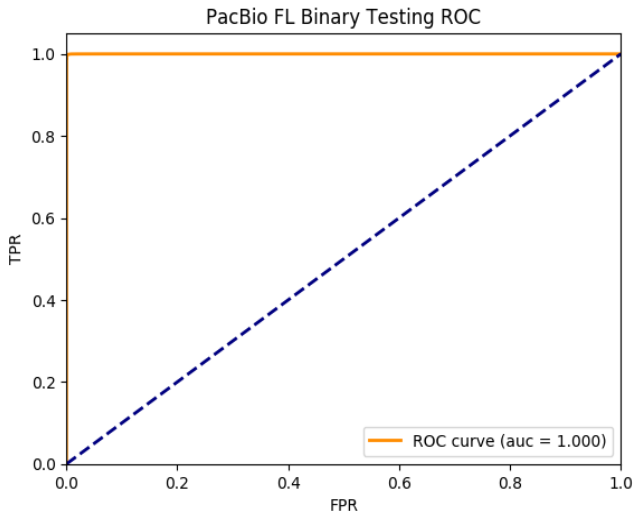

Figure B.2: A. Oxford FL Binary Class ROC on Testing (held-out) data

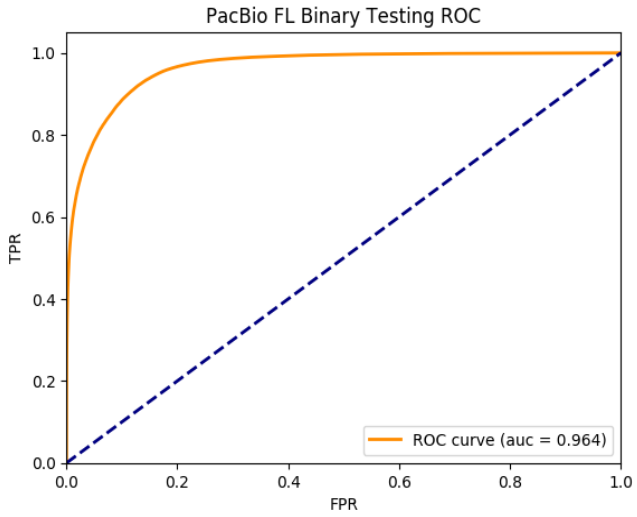

Figure B.3: B. Oxford non-FL Binary Class ROC on Testing (held-out) data

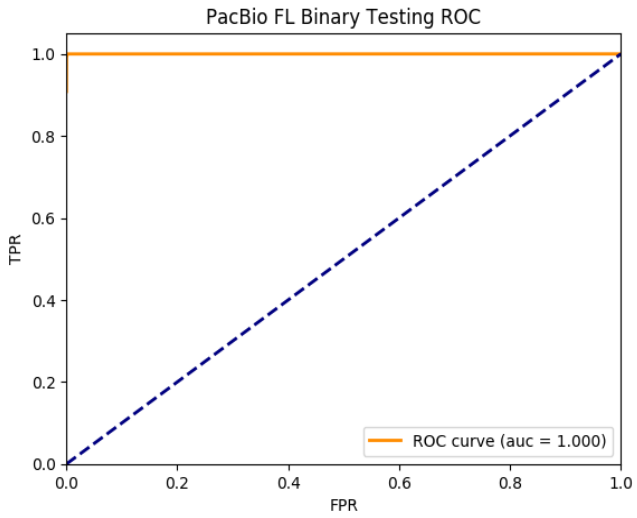

Figure B.4: C. PacBio FL Binary Class ROC on Testing (held-out) data

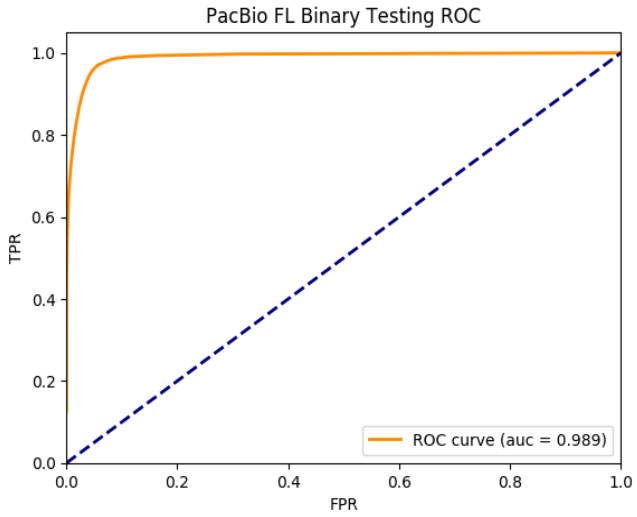

Figure B.5: D. PacBio Non-FL Binary Class ROC on Testing (held-out) data

Table B.2: NA12878 Alignment Class Recall. Totals in the table are per-category and based on the total number of alignments that overlapped a transcript with that class label form the training data.

Dataset	Total Reads Categorized	Problem-free	Any error	Recurrent error	Novel
Oxford full length	2,749,480 / 5,284,512	89% (2282118 / 2552987)	17% (453539 / 2646694)	18% (12421 / 70490)	10% (1402 / 14341)
Oxford fragment	2,406,066 / 5,492,870	44% (1519189 / 3468504)	62% (883498 / 1418051)	1% (3221 / 292757)	0% (158 / 313558)
PacBio full length	228,317 / 261,944	98% (224685 / 228633)	12% (3493 / 29760)	3% (72 / 2072)	5% (67 / 1479)
PacBio fragment	120,756 / 285,799	48% (115026 / 240716)	21% (5399 / 26276)	38% (238 / 630)	1% (93 / 18177)

Table B.3: NA12878 Alignment Class Precision. Totals in the table are per-category and based on the total number of alignments that were predicted to have that class label. Totals are the same between precision and recall and are repeated for convenience.

Dataset	Total Reads Categorized	Problem-free	Any error	Recurrent error	Novel
Oxford full length	2,749,480 / 5,284,512	51% (2282118 / 4459215)	67% (453539 / 675162)	34% (12421 / 36293)	1% (1402 / 113842)
Oxford fragment	2,406,066 / 5,492,870	62% (1519189 / 2434882)	29% (883498 / 3043506)	23% (3221 / 13835)	24% (158 / 647)
PacBio full length	228,317 / 261,944	88% (224685 / 254032)	51% (3493 / 6835)	42% (72 / 171)	7% (67 / 906)
PacBio fragment	120,756 / 285,799	83% (115026 / 137759)	11% (5399 / 48902)	0% (238 / 97771)	7% (93 / 1367)

Table B.4: Intron Chains in Annotation [exact (fuzz) percent matching]

Annotation	Total Intron Chains	Illumina NA12878	Illumina SKBR3	Oxford NA12878 (OX-RAW)	Oxford NA12878 FLAIR	PacBio NA12878 (PB-RAW)	PacBio NA12878 FLAIR	PacBio SKBR3 (PB-SKBR3)
Gencode V29	199,381	**17.0% (29.8%)**	12.0% (22.7%)	**35.9% (44.1%)**	15.2% (25.9%)	17.5% (26.2%)	7.1% (14.2%)	27.2% (37.3%)
Union of Annotations	1,098,511	**15.4% (26.3%)**	12.4% (20.7%)	**36.2% (44.2%)**	13.0% (22.0%)	19.0% **(26.2%)**	7.6% (12.7%)	28.1% (38.3%)

Table B.5: Improvement of intron-chain matches from problem free predictions

Dataset	**Intersection** of FL & Fragment Problem Free Predictions vs. **Union of Annotation**	**Union** of FL & Fragment Problem Free Predictions vs. **Union of Annotation**
NA12878 Oxford vs. Annotation	82.7% (97.5%)	82.8% (96.9%)
NA12878 Pacbio vs. Annotation	57.2% (86.7%)	46.3% (82.0%)
SKBR3 Pacbio vs. Annotation	31.3% (67.3%)	28.3% (64.6%)
Annotation vs. NA12878 Oxford	**19.9% (25.9%)**	**33.2% (40.7%)**
Annotation vs. NA12878 Pacbio	13.7% (20.5%)	**18.8% (26.0%)**
Annotation vs. SKBR3 Pacbio	22.4% (33.7%)	27.9% (38.1%)

Appendix C

Additional Details of Snaptron

C.1 Analyses

For the SSC analysis the exon from the SRGAP2B gene was not able to be lifted over from GRCh37 to GRCh38 and was therefore not analyzed. Scripts and data needed to reproduce the three analyses presented in this study are available at:

- https://github.com/ChristopherWilks/snaptron-experiments/tree/feb2017_manu_rc1

The exact version of Snaptron used for the analyses presented in the paper are available at:

- https://github.com/ChristopherWilks/snaptron/tree/feb2017_manu_rc1

Christopher Wilks

12916 Fork Road, Baldwin, MD, 21013, (831) 239-3879, cwilks3@jhu.edu

Interested in working on large scale computational processing of genomics-related data to benefit human health and biomedical research

Major Projects

10+ years working on bioinformatics related software & system projects, with substantial experience and focus working with large data projects including transferring & processing of 100,000's of genomic sequence files efficiently (BAMs/FASTQs):

- ◇ Codeveloper of the recount3 resource and Monorail workflow: aligning and summarizing coverage from over 770,000 RNA-seq runs from human & mouse including cancer patient data (TCGA) and multi-tissue normal transcriptomes (GTExV8) across heterogeneous compute environments including HPCs (Slurm) and AWS, `https://github.com/langmead-lab/monorail-external`
- ◇ Primary developer of Snaptron: a custom query engine to quickly search & filter for transcriptomic coverage of genomic regions at gene, exon, splice junction, and base pair resolutions across approximately 100,000 RNA-seq runs, `https://github.com/ChristopherWilks/snaptron`; `https://github.com/ChristopherWilks/snaptron-experiments`
- ◇ Primary developer of Megadepth: a multi-threaded program to efficiently extract coverage summaries from BAM & BigWig files and a core part of the Monorail workflow, `https://github.com/ChristopherWilks/megadepth`
- ◇ Was one of the primary software engineers and the bioinformatician at the award-winning, NIH-funded petabyte-scale cancer genomics repository, the Cancer Genomics Hub (CGHub) [1]
- ◇ Worked collaboratively with an international team of cancer researchers and engineers to implement and run a DNA-seq whole-genome alignment pipeline for the re-analysis of ~2000 whole-genomes from ~1000 patients as part of the early Pan-Cancer Analysis of Whole Genomes (PCAWG) project
- ◇ Primary developer of a BAM file repair and concatenation pipeline (in C and Python) leveraging the Samtools C API, used to fix hundreds of terabytes of pediatric cancer genomics files, which were subsequently released to the cancer research community as part of the TARGET/CGHub projects
- ◇ Worked with molecular biologists as the primary developer on a structural genomics annotation project resulting in five public gene annotation releases (at the former The Arabidopsis Information Resource [TAIR] group [2])

Technical & Professional Skills

- ◇ Multiple years of experience with Python, Perl, C/C++, Bash, Java, and SQL; some experience with development tools (GDB, Valgrind, gprof, and Intel VTune); some experience with R/Bioconductor.
- ◇ Knowledge of, and day-to-day working experience with cluster/grid computing (Slurm, SGE), AWS, database applications, Linux, Solaris, and Windows servers and desktops
- ◇ Knowledge of, and day-to-day working experience with container technology using Docker & Singularity
- ◇ Experience applying Random Forests to predicting spliced-alignment errors in PacBio Iso-Seq & Oxford Nanopore long read sequences and improving gene annotation in Malaria (*P. falciparum*) in an academic setting

- ◇ Experience with network transfer performance optimization over WANs, network security auditing, and protocol debugging
- ◇ Understanding of security concepts such as asymmetric cryptography and hashing algorithms, and experience debugging SSL-based applications
- ◇ Education in software engineering with an emphasis in design patterns and proper coding techniques, including experience with various version control software, test frameworks, debugging tools, and performance profiling
- ◇ Knowledge of the core concepts of molecular biology, primarily the central dogma, including DNA, RNA, the transcription process, translation into proteins, splicing, and non-coding RNAs, among other related processes
- ◇ Knowledge of and experience with biologically related databases/tools/algorithms including Samtools, Picard, GenBank, UniProtKB/Swiss-Prot, InterProScan, the NCBI-BLAST tool suite, TargetP, TransmembraneHMM (TMHMM), Genewise, FASTA, GeneSeqer, GMAP, Blat, Sim4, BioPerl, BioJava, Dot Matrices, Dynamic Programming, Global & Local Sequence Alignment, Scoring Matrices, Multiple Sequence Alignment, Fragment Assembly, and Gene Browsers

Talks

- ◇ **Snapcount: rapid and flexible querying of over 70,000 gene, exon, and splice junction expression summaries**
  BioC 2019: Where Software and Biology Connect (June 2019)
- ◇ **Innovations in Networking Award for High Performance Research Applications:**
  A Network Transfer Protocol for Genomic Data
  Corporation for Education Network Initiatives in California (CENIC) conference (2013) [3]

Selected Publications and Posters

- ◇ **ASCOT identifies key regulators of neuronal subtype-specific splicing.**
  Jonathan P Ling, **Christopher Wilks**, Rone Charles, Patrick J Leavey, Devlina Ghosh, Lizhi Jiang, Clayton P Santiago, Bo Pang, Anand Venkataraman, Brian S Clark, Abhinav Nellore, Ben Langmead, Seth Blackshaw.
  Nature Communications (2020): doi:10.1038/s41467-019-14020-5
- ◇ **Snaptron: querying splicing patterns across tens of thousands of RNA-seq samples.**
  Christopher Wilks, Phani Gaddipati, Abhinav Nellore, and Benjamin Langmead.
  Bioinformatics (2018): doi:10.1093/bioinformatics/btx547
- ◇ **Flexible expressed region analysis for RNA-seq with derfinder.**
  Leonardo Collado-Torres, Abhinav Nellore, Alyssa C. Frazee, **Christopher Wilks**, Michael I. Love, Ben Langmead, Rafael A. Irizarry, Jeffrey T. Leek, and Andrew E. Jaffe.
  Nucleic Acids Research (2016): doi:10.1093/nar/gkw852
- ◇ **Rail-RNA: Scalable analysis of RNA-seq splicing and coverage.**
  Abhinav Nellore, Leonardo Collado-Torres, Andrew E. Jaffe, José Alquicira-Hernández, **Christopher Wilks**, Jacob Pritt, James Morton, Jeffrey T. Leek, and Ben Langmead.
  Bioinformatics (2016): doi:10.1093/bioinformatics/btw575

- ◇ **Rail-dbGaP: analyzing dbGaP-protected data in the cloud with Amazon Elastic MapReduce.**
  Abhinav Nellore, **Christopher Wilks**, Kasper D. Hansen, Jeffrey T. Leek, and Ben Langmead.
  Bioinformatics (2016): doi:10.1093/bioinformatics/btw177

- ◇ **The Cancer Genomics Hub (CGHub): overcoming cancer through the power of torrential data.**
  Christopher Wilks, Melissa S. Cline, Erich Weiler, Mark Diehkans, Brian Craft, Christy Martin, Daniel Murphy, Howdy Pierce, John Black, Donavan Nelson, Brian Litzinger, Thomas Hatton, Lori Maltbie, Michael Ainsworth, Patrick Allen, Linda Rosewood, Elizabeth Mitchell, Bradley Smith, Jim Warner, John Groboske, Haifang Telc, Daniel Wilson, Brian Sanford, Hannes Schmidt, David Haussler, and Daniel Maltbie.
  Database (Accepted 26 August 2014) Vol. 2014: article ID bau093; doi:10.1093/database/bau093

- ◇ **Comprehensive molecular portraits of human breast tumours.**
  Cancer Genome Atlas Network.
  Nature (04 October 2012) 490(7418), 61-70. doi:10.1038/nature114122012

- ◇ **The Arabidopsis Information Resource (TAIR): Gene Structure and Function Annotation.**
  David Swarbreck, **Christopher Wilks**, Philippe Lamesch, Tanya Z. Berardini, Margarita Garcia-Hernandez, Hartmut Foerster, Donghui Li, Tom Meyer, Robert Muller, Larry Ploetz, Amie Radenbaugh, Shanker Singh, Vanessa Swing, Christophe Tissier, Peifen Zhang and Eva Huala.
  Nucleic Acids Research (2008) 36 (suppl 1): D1009-D1014. doi: 10.1093/nar/gkm965

- ◇ **A fast shotgun assembly heuristic.**
  Christopher Wilks, Sami Khuri.
  Computational Systems Bioinformatics Conference, 2005. Workshops and Poster Abstracts. IEEE, 122-123. doi:10.1109/CSBW.2005.7

Education

Rising 6th year PhD student in Computer Science at Johns Hopkins University, 4.0 GPA, projected graduation by end of year 2020 (started Fall 2015).
M.S. in Computer Science from University of California, Santa Cruz (UCSC) 2011, 4.0 GPA
B.S. in Computer Science from San Jose Sate University, Minor in Mathematics 2004, Magna Cum Laude

- ◇ Master's Thesis: "Exploiting MAINE/FAIRE Experimental Data for Gene-Finding in *Plasmodium falciparum* Using Random Forests"
- ◇ Graduated with undergraduate departmental honors for an independent study project aimed at implementing a method in Java for discovering non-coding RNAs
- ◇ One of six students (both graduate and undergraduate) chosen to represent SJSU at the California State University (CSU) research competition in 2005 at Sacramento, CA
- ◇ 2nd place winner at the CSU state research competition in 2005 for the project "A Fast Heuristic for DNA Fragment Assembly" in the Engineering and Computer Science, Undergraduate level
- ◇ UNIX Systems Administration Certification from San Jose Sate University

Work History

ModMab Therapeutics (MedGenome), Foster City, California
Jan 2019 - August 2019: Research Contractor (P/T)

◇ Aggregated gene expression with mutation data for specific gene targets sourced from TCGA, CCLE, and/or GTExV6 samples

Genentech, South San Francisco, California
May 2017 - August 2017: Research Intern (gRED)

◇ Adapted SGSeq [4] alternative splice event predictions to be input to five existing R/Bioconductor differential expression packages originally intended for transcript/exon level data

◇ Benchmarked the previous item's results using synthetic & real RNA-seq samples for accuracy and computational performance

◇ Developed an R package and workflow around this work (not published)

Cancer Genomics Hub (CGHub), School of Engineering, UCSC
August 2011 - June 2015: Software Engineer and Bioinformatician (Programmer/Analyst III)

◇ Primary data transfer and software engineer facilitating the download of approximately 20 Petabytes of cancer genomes over 2+ years by the cancer research community (measured by initiated downloads only)

◇ Primary engineer tasked with repairing and releasing approximately 1000 pediatric cancer genome files for the Therapeutically Applicable Research to Generate Effective Treatments (TARGET) project

◇ Primary engineer assigned to setup, upgrade, and maintain the Genomic Network Operating System (GNOS) including GeneTorrent; a 3rd party metadata tracking and transfer software used to manage over 1.5 Petabytes of cancer genomics in more than 70,000 files

◇ Contributed to the ongoing design and testing/debugging of GNOS/GeneTorrent software system employed to upload, download, and track cancer genomics files at CGHub

◇ Primary engineering liaison between UCSC and the engineering subcontractor developing GNOS

◇ Technical representative of CGHub at various conferences from 2011 through 2014 which included a technical talk, multiple workshops, and poster sessions

◇ Continually acted as engineering representative liaising with the National Cancer Institute and Leidos Biomedical Research management of the CGHub contract and various 3rd party users and submitters to CGHub

Center for Biomolecular Science and Engineering, School of Engineering, UCSC
February 2011 - August 2011: Software Engineer/Bioinformatician (Programmer/Analyst III)

◇ Worked as a software developer supporting and contributing to the analysis of cancer genomics data coming from the largest cancer genomics sequencing project with ~10 thousand patients sequenced one or more times to date, The Cancer Genomics Atlas (TCGA).

◇ Contributed to the creation of a prototype RNA-seq alignment and transcript expression pipeline using Tophat and Cufflinks

◇ Responsible for running and reviewing results of an in-house DNA Variant caller over several hundred cancer patients genomes (CNV, SNV, structural)

◇ Prototyped a cancer genomics file browser using the Python Web framework Django which led to the creation of the official CGHub Data Browser

The Arabidopsis Information Resource (TAIR), Plant Biology Dept.,
at The Carnegie Institution of Washington, Stanford, California
June 2004 - June 2005: Intern
June 2005 - Jan. 2011: Bioinformatics Software Developer

- Primary developer on a genome wide gene annotation project where 30% (~9,000 genes) of the genome was updated and annotated to incorporate additional experimental evidence and biologically driven corrections
- Primary developer on the RNA-Seq analysis process used in the annotation project above, worked with Tophat, Cufflinks, Supersplat, TAU and other programs in the project
- Main programmer for collaborative web services for a locus based gene information retrieval project using the Tomcat servlet server with the Axis engine utilizing a JDBC interface to a Sybase database as the backend
- Primary developer for the interface between a genomic coordinates database (MySQL) and an automated annotation pipeline using XML-RPC to communicate between Perl and a Java Data Objects (JDO) object-to-relational database mapping structure
- Primary developer for an interface between Apollo, a genome annotation tool, and the JDO interface mentioned previously, both written in Java
- Worked with a biological curator to extend an automated pipeline for annotating genes to include potential non-coding RNAs
- Involved in the design of a proprietary database schema to store mapped biological objects (cDNAs,ESTs, and tDNAs)
- Contributed to the design, enhancement, and maintenance of an object-oriented data layer for the aforementioned database in Java using JDO technology
- Repaired and maintained a set of Perl scripts for the assignment of biologically significant DNA sequences onto the Arabidopsis genome (cDNAs, ESTs, Polymorphisms, and tDNAs)
- Various stand alone scripts relating to cDNA, EST, and tDNA genomic assignments, all written in Java and interfaced with JDO
- Wrote and modified CGI scripts in Perl and PHP for visualization of different biological structures
- Installation and administration of MySQL, Apache, and Perl on multiple systems

Gadzoox Networks, Inc., San Jose, California
January 2000 - October 2001, August 2002 - November 2002 IT Engineer (P/T)

- Worked as primary administrator for enterprise-wide backup and restore for the HQ site
- Responsible for configuration of the storage area network (SAN)
- Primary administrator of anti-virus software for the HQ site, including servers, e-mail, and desktop machines
- Project lead for domain migration from NT 4.0 Lanman services to Windows 2000 Active Directory
- Project lead on migration from Exchange 5.5 to Exchange 2000
- Backend Exchange E-mail server administrator

- Internal ISO Auditor
- Desktop/Helpdesk support person
- Print Administrator

[1]: https://cenic.org/news/uc-santa-cruz-cancer-genomics-hub-wins-innovations-in-networking-award-for
[2]: https://www.arabidopsis.org/
[3]: https://www.youtube.com/watch?v=K_HRO1WppaQ
[4]: Goldstein, L. D., Cao, Y., Pau, G., Lawrence, M., Wu, T. D., Seshagiri, S., & Gentleman, R. (2016). Prediction and quantification of splice events from RNA-seq data. PloS one, 11(5), e0156132.