Optimizing Existing Models of RNA Sequencing: A Novel Infrastructure to Advance the Timeline of Gene Sequencing Methods

Abstract

The emergence of new gene-sequencing technologies creates the potential for more personalized treatment options available to patients. To contribute to the effort of identifying these treatments in a timely manner, we helped create an RNA sequencing pipeline that analyzes data and produces outputs efficiently, helping push for higher accuracy in determining proper treatments for cancer patients by reducing turn around time. We containerized the several genomic tools used in the pipeline with a portable structure called Docker. This structure creates virtual environments that can be run on practically any operating system, allowing our pipeline to not only be used in standardized hospitals, but also in third-world countries where specifics are not thoroughly documented. TOIL, a batch job scheduler and resource management system, was developed in tandem with the pipeline, allowing many samples to be run in parallel on clusters set up in the commercial cloud. A variety of tools were used ranging from simple searching and sorting algorithms to post alignment quality control. All of these tools work together to produce several useful outputs for researchers to utilize in their search for optimal treatment paths. With the combination of Docker, TOIL, and each handpicked tool, the RNA sequencing pipeline's performance exceeds that of the typical methods used in the bioinformatics world. As a result, this pipeline will be launched in hospitals to not only cut costs but also advance the timeline in which patients are diagnosed with meaningful treatments.

Optimizing Existing Models of RNA Sequencing: A Novel Infrastructure to Advance the Timeline of Gene Sequencing Methods

Executive Summary

RNA sequencing is a method of taking in large amount of genetic data, analyzing it, and outputting information about the samples that can be used towards medical research. Scientists in the field of cancer research rely on software analytics to determine efficient treatments for a wide range of cancers. However, the effectiveness of these treatments varies greatly; some are matched well, while others do not work and end up costing extensive amounts of time and money. Moreover, current RNA sequencing pipelines cost an invaluable amount of time to process data – time that could be spent treating the patient. We helped create a pipeline that incorporates current gene sequencing tools that work automatically to speed up the production of the important data scientists to use in their search for viable treatment procedures for patients. The pipeline uses a cloud based job scheduler called TOIL and a new software structure called Docker. TOIL and Docker allow the pipeline to be reliable on large clusters, less resource intensive, and able to produce outputs in a timely manner when compared to other pipelines that, again, churn out exactly the same data, but in a significantly greater amount of time. Both TOIL and Docker automate the process allowing computing clusters in any country to run it, without highly trained personnel.

1 Introduction

In the past fifty years, the survival rate for childhood cancer patients has increased from 20% to 80% (Gatta et al.). However, while this statistic points towards an improving trend for the viability of forthcoming cancer treatments, the increase is primarily due to treatments which lead to harmful, costly, and toxic side effects (Russel et al. 2013). These aggressive treatments are pursued even through potential misdiagnoses (Hoadley et al. 2014). However, along with the survival rate, costs for treatments are also on the rise. For example, average bone marrow transplants now go upwards of $200,000 (Matthes-Martin et al. 2012). The new era of computing allows hospitals to operate on big data sets to identify and offer precise treatments for a newly diagnosed patients. These genome analysis procedures are not just more humane, but much cheaper overall, as they hone in on the correct treatment, saving the trouble of misdiagnoses and the loss of money they entail.

DNA and RNA sequencing is being used more frequently to find treatments for children with cancers (Tirode et al. 2014). However, this technique is not being used to its full potential because of the lack of a streamlined and efficient pipeline. Thus, the creation of an *RNA-sequencing pipeline* (or RNA-seq pipeline) is an important step toward increasing the success rate of precision treatments.

The pipeline discussed in this paper is a drastically improved model based on previous RNA-seq pipelines. Instead of receiving pertinent data on upwards of a week, our pipeline allows for data sets to be operated on in just a few days, while providing the same big data functionality as other pipelines. This pipeline revolutionizes the industry by cutting costs, thereby saving millions of dollars per year that would have incurred a significant time penalty and a larger percentage of misdiagnosed treatments. Even more importantly, this pipeline has the potential to

saves lives by allowing for a quick and accurate diagnoses of patients awaiting treatments at hospitals.

The paper is organized as follows: Section 1 provides background information about the gene sequence pipeline and the incorporated theories. Section 2 details the construction of the pipeline using the overarching framework, *Docker*. The tools built into the pipeline are examined independently in Section 3. Section 4 overviews the methods used in passing data through the pipeline. In Section 5, we performed a real-life test of the pipeline and thoroughly investigated the results to confirm that the results are the same as a manual transformation of the data. Finally, we express our concluding thoughts and ideas for future work in Section 6.

1.2 RNA Sequencing

In the recent years, RNA sequencing has become a major part of genetic analysis procedures in laboratories around the world, as it provides a simpler, more efficient solution to profile and formulate gene sequences than traditional gene expression microarrays (Mardis et al. 2009). The majority of RNA sequencing is carried out by using messenger RNA, or mRNA, to synthesize complementary DNA, or cDNA (Parkhomchuk et al. 2009).

While RNA sequencing is still a fairly new concept, it is extensively tested and has procured results from many scientists in the gene sequencing sector. For example, the method's adaptability has been used to find rare sections of genetic sequences and even find entirely new genes. RNA sequencing is turning out to be a valuable tool for many scientists, as the analysis of the structure and function of RNA offers a wealth of quantitative and qualitative data to be inspected. For example, many cancer tumors can be identified and distinguished by the subtle differences in genes observed from sequencing RNA (Wang et al. 2009). Current pipelines can be improved by reducing the time taken to operate on large datasets.

1.3 Desired Impact

The ultimate purpose of the project we worked on was to provide a streamlined, efficient, manageable pipeline to the CKCC (Child Kids Cancer Comparison), an organization dedicated partly to raising awareness and usage of gene sequencing methods to find the right treatments for individual cancer patients. The pipeline expedites the timeline and lessens the cost of RNA-sequencing procedures, aiding in the overall goal of increasing the effectiveness of genome analysis from about 10% to at least 20%. CKCC is a driver project in which all of the patients are still alive; thus, turn around time is very critical. Hospitals that will eventually make use of this pipeline currently have major issues with non-streamlined methods that take massive amounts of time to provide the same information.

2 Docker

Containerizing the tools used in the pipeline required a utility that allows for apps to be independent of bulky operating systems. Docker is a lightweight option to "containerize" desired applications into downloadable packages that have all required files and dependencies needed to run the tool (Felter et al. 2014). This method eliminates traditional virtual machine related inefficiencies and offers the user an easier way to debug problems without worrying about the nuances of different operating systems. Traditional pipelines do not implement a Docker based structure, resulting in reduced scalability and portability.

2.1 Docker Information

Containerizing an application is a fairly straightforward process, requiring the creation of an image (an immutable snapshot of a container) through simple Docker commands procedurally executed in a Dockerfile. A Dockerfile contains a mix of Unix and Docker code to download, update, and install required dependencies from specified repositories. The Dockerfile is used to

build an image of the desired application, after which the image is booted as a container that has virtual machine-like functions.

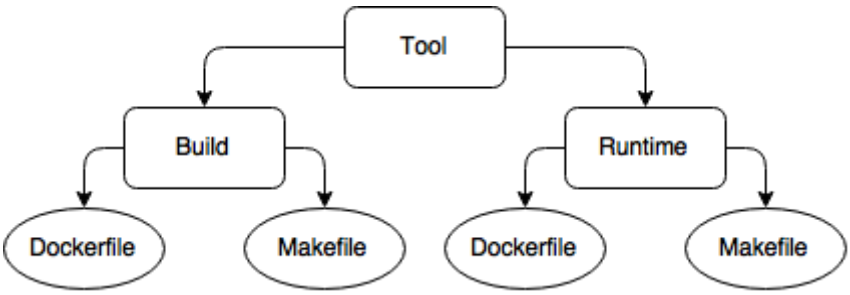


Figure 1: A hierarchical diagram showing the locations of files with respect to build and runtime directories. These two directories separate files that pertain either to the construction or accession of dependencies from files that actually run the tool.

We followed concrete guidelines for each containerized tool used in the RNA sequencing pipeline to ensure uniformity, reproducibility, and optimal usage in the pipeline script. Another file needed in the containerization of the software, the Makefile, allows a user to easily build tools from source if they desire. It should be noted that all Docker tools should use the binary image which ensures a degree of reproducibility that cannot be guaranteed by building a tool from source.

Because a standardized workflow is not utilized by Docker itself, the guidelines that we followed are specifically tailored to our purposes and work best with the library of software used in the RNA-seq pipeline. Therefore, not all Docker projects are the alike and might be formatted in a completely different way than ours was.

2.2 Docker Syntax

Dockerfiles specify the *base image* (image that the said Docker container is built upon) and the commands to be run for the creation of new image. Each command is preceded by the

word “RUN.” Once created, the Docker image is shared on an online platform called *Dockerhub*, allowing our containerized toolset to be easily accessible around the world (Rybicki et al. 2015).

2.2.1 Pseudo Code

The framework of the pseudo code for a tool that needs to be containerized is as follows. This code would go into the tool’s corresponding Dockerfile.

```
//Base Image (in this case: ubuntu)
FROM ubuntu

//Update existing sources and download dependencies
RUN apt-get update && apt-get install -y [dependencies]

//Clone git repositories, if any (similar to downloading dependencies)
RUN git clone [link to repository]

//Set the current working directory to the cloned folder
WORKDIR [tool name]

//Follow instructions to install the tool that was just containerized above
RUN make

//Entrypoint that is launched following the Docker container's creation
ENTRYPOINT ["tool name"]
```

Figure 2: The images displays the standard Dockerfile entered into a text editor. Each line is accompanied by a comment demonstrating its usefulness and necessity.

The Dockerfile contains a list of instructions executed when the Docker container is made. Some steps pertain to acquiring dependencies required by the containerized software, while others simply change working directories. An *entrypoint* command is utilized in every container to run the main method on container’s launch. As a result, the hassle of navigating through the virtual directories is non-existent.

2.2.2 Docker Conclusion

Ultimately, Docker provides a lightweight, easy to implement alternative to traditional virtual machines, assisting in the overall goal of creating a stable, portable gene sequencing pipeline that is also easily run on large clusters. One of its major benefits is the ability to run the pipeline on all operating systems. Docker also comes with many security advantages, as a software's administrator is allowed discretion on the directories users can access, keeping patient data in the pipeline safe and secure.

3 Tools

The RNA-seq pipeline consists of many different tools that are used in conjunction to analyze and format the data that is inputted. Each tool is tailored to a certain portion of gene analysis. The following tools were containerized according to the methods explained above to generate a fully portable gene analysis pipeline. These tools are selected in a specific order explained in Section 4 that makes the entire pipeline more streamlined than others in the field. The automation of these tools was an important part of making our pipeline faster than those that already exist. The details of each containerized tool are mapped out below.

3.1 Samtools

Samtools is a set of tools based on the SAM (Sequence Alignment + Map) format. The SAM format can hold immense nucleotide sequences in a relatively small file that can be read from and indexed easily. Samtools enables analysts to search, sort, index, merge, and do much more to the SAM format files passed through. It serves as a preliminary step to further, more complicated research and analysis procedures (Li et al. 2009).

3.2 Picard-tools

Similar to Samtools, Picard-tools is another tool that deals with the SAM format (even offering support for the binary SAM format, BAM. It allows for the manipulation of SAM format files, such as sorting, searching, etc. Essentially, Picard-tools is a Java based SAMTools (Niemenmaa et al. 2012).

3.3 Mapsplice

The importance of mapping splice locations in RNA-sequencing data cannot be overstated. Mapsplice allows for the accurate detection of splice locations in an efficient manner. Mapsplice could be used in this project as it works with short files as well as incredibly long ones. The goal of Mapsplice is to be as accurate and reactive to splice junctions as possible. (Wang et al. 2010).

3.4 Ubu

Ubu is another Java based tool, mainly dealing with conversions and formatting of files such as SAM, BAM and FASTQ. There is a splice computing feature and SAM file statistic counter built in as well. Thus, Ubu is used primarily as an intermediate step between more major algorithmic ones.

3.5 Bedtools

Bedtools is another toolset similar to Samtools and Picard-tools, specifically built for complex mathematical algorithms performed on set based gene data. There are some alternative types of files that are supported, such as BED and GFF, as well as others such as BAM. Some major functions that are present in bedtools are *intersect*, *count*, *merge*, *complement*, and *shuffle*, each of which modify the given data. These tasks can be compounded to allow for the

manipulation of a given gene sequence for easy qualitative and quantitative analysis (Quinlan et al. 2010).

3.6 RSEM

RSEM is a RNA-seq tool commonly used for quantifying gene isoform abundances and when paired with *De novo transcriptome assembly*, it does not need a reference genome. In the script, *rsem-calculate-expression* is used to both handle the alignment of reads against a reference genome, and the calculation of relative abundances via the *Bowtie* alignment program (Bowtie citation). RSEM has comparable performance to other quantification software, but is a vastly superior alternative due to its ability to run without a reference genome. As a result, RSEM has enabled us to perform cost efficient quantification for RNA-seq, currently a very expensive procedure (Li et al. 2011).

4 Methods

4.1 Prerequisites

In order to maximize the pipeline's efficiency, certain universal dependencies are required to be used in most steps. The necessary dependencies are: (1) cURL, (2) Docker, (3) Samtools, (4) unzip, (5) TOIL.

cURL is an automated server data transferring utility designed to work with popular protocols (Stenberg et al.). In the pipeline, the download steps use cURL to extract pertinent data from desired servers and move to a *filestore*.

Docker is the major factor in the pipeline's portability. It's numerous advantages and contributions to the pipeline's speed were explained previously.

Samtools provides many options to configure and sort gene data. In the pipeline, Samtools is used specifically for its *sort_bam_by_reference* function to prepare data.

Unzip is essentially a file un-archiving dependency that equips the user with methods to un-package zip files in a controlled manner (Goatley et al. 2009). Used in combination with TOIL, Unzip allows for organized space allocation and file management.

4.2 TOIL

In order to alleviate the resource toll of the script based on the unique cluster that it is run on, we incorporated TOIL, a Python based management software developed in the lab for clusters that renders running recursive and dynamically scheduled computations straightforward. This software utilizes a direct acrylic graph (DAG), where every method is organized into a node. TOIL allows the user to specify jobs to be run as children nodes or as follow-ons, making the script run as many jobs as possible. TOIL also gives users the ability to efficaciously allocate RAM, computing resources, and storage. Other pipelines have reduced speed and lack automation because they do not take advantage of large scale cluster management tools.

4.3 Visual Pipeline Overview

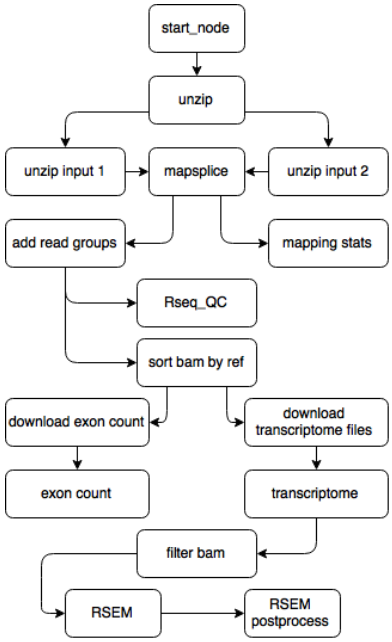


Figure 3: An overarching nodal representation of the pipeline flow. Each node represents method calls or branching helper methods that all combine to process, and output transformations of the inputted FASTQ files.

5 Test Case

In the following subsections, a real-life test case will be examined. The patient data is gathered from a hospital; however, due to patient confidentiality agreements, none of the data shown can be detailed enough to explicitly link back to the patient. The main purpose of the patient data is to demonstrate a real world usage of the RNA-seq pipeline and showcase its consistency with other methods. As mentioned earlier, every method is strung together to create an essentially automated version of the pipeline. As a result, bioinformaticians do not have to deal with the manual transformation and passing of data.

5.1 Thorough Analysis

start_node: The start_node process instantiates all relevant variables unique to the sample that is passed in; these variables are used throughout the rest of the pipeline. The sample is downloaded as the result of a child process called *download_encrypted_file*, which grabs FASTQ files from a specified path in a overarching server and adds them to the global *filestore*. This method adds *unzip* as a child node.

unzip: *unzip* uses the *Unzip* dependency to launch two parallel processes that unarchive the downloaded FASTQ files using the *zcat* framework. This method adds *mapsplice* as a subprocess and updates the global *filestore* with the recently obtained FASTQ files while also deleting the previous set.

mapsplice (mapping stats): *mapsplice* retrieves sample 1 and sample 2 (FASTQ files stored in the global *filestore*) and starts the Docker container for Mapsplice. After updating the global *filestore*, the tool then launches *mapping_stats* as a sub node. *Mapping_stats* again initializes a Docker container and outputs three files (stats2.txt, stats_all.txt, and mapping.tab)

before finally re-updating the global *filestore*. In parallel to *mapping_stats*, the *mapsplice* function adds another child node called *add_read_groups*.

add read groups: This function launches a Docker container running Picard-tools which is used to add or replace read groups in the sample FASTQ files. The *AddOrReplaceReadGroups* method, which is in the Picard-tools container, is called to consolidate all separate read groups in the input files and clones all reads to a singular read group in an output file.

Rseq_QC: The *rseq_qc* function launches a Docker tool called *QC*, a separate quality control set of tools that generate graphs which are discussed in the Graphs section.

sort bam by ref: The method *sort_bam_by_reference* opens a subprocess of Samtools called *cat*, which combines different BAM files into one and creates an output called *sort_by_ref.bam*. This output has prepared data obtained through sorting inputted bam files, containing gene sequencing data, which allows for alignments for reference genomes. After transforming the bam files, the function updates the *global filestore* and launches exon and transcriptome as parallel, child nodes.

exon count: The *exon_count* function launches a Docker container loaded with Bedtools, and calls the *coverage* method that is built in. This Bedtools subtool is used to calculate the coverage of aligned gene sequences. Finally, the *exon_quant* file is generated, outputted to the user directory, and then pushed to the *global filestore*.

transcriptome: In order to prepare the data for the *filter_bam* step, the transcriptome function launches a docker container of Ubu, a java based tool used for the translation of files. In this Ubu container, the transcriptome.bam file is created by translating the previous bam file from genome coordinates to transcriptome coordinates. Then, the *global filestore* is updated to

contain the new transcriptome.bam file. Finally, this node launches the *filter_bam* function as a subprocess in the job structure.

filter bam: Takes in *transcriptome.bam*, calls Ubu, and returns *filtered.bam* as an output. The output is a filtered version of the input with all reads that do not align removed. As usual, after outputting *filter.bam* the function updates the *global filestore*; subsequently, RSEM is initialized.

RSEM: The RSEM function retrieves the filtered.bam and unzips the rsem reference files to pass into an initialized docker container of RSEM. From this container, preliminary tab files are generated that contain many statistics discussed in the Outputs section. These preliminary tab files are used as inputs for the RSEM postprocess step, which is launched as a sub node.

RSEM postprocess: Takes in *rsem_isoform.tab* and *rsem_gene.tab* and compiles a multitude of tab files that contain relevant genetic analysis information. These tab files are discussed in the Outputs section.

5.1.1 Outputs

The RSEM postprocess step is the final point of the pipeline in which outputs are produced and moved to the user's directory for analysis. Used independently each tool would produce a segment of the final outputs; however, this data would take much longer to manifest with traditional pipelines/methods. The following outputs are provided:

- Isoforms results: This file contains isoform level expression estimates
- Isoforms raw counts: This file contains baseline overlapping isoforms; For use with Isoforms norm counts.
- Isoforms norm tpm: This file contains a TPM (transcripts per million). TPM refers to the proportion of transcripts within the entire RNA sequence.

- Isoforms norm fpkm: This file contains an fpkm (fragments per kilobase of transcript per million reads) for the isoforms relative to the isoform raw fpkm.
- Isoforms norm counts: This file contains the number of overlapping isoforms relative to the isoform raw counts.
- Genes results: This file contains gene level expression estimates
- Genes raw count: This file contains the number of overlapping genes
- Genes norm tpm: This file contains TPM for Genes.
- Genes norm fpkm: This file contains fpkm for Genes.
- Genes norm counts: This file contains a normalized count of the genes relative to the raw count.
- Exon quant: This file contains the Bedtools coverage method output.

5.1.2 Graphs

The following graphs are all compiled from the *Rseq_QC* output data and offer valuable information about the samples fed through the pipeline. *Rseq_QC* is used mainly to error check operations conducted when sequencing large data sets, as even a small mistake or miscalculation can bias an entire set. Using the Rseq_QC method independently would produce the exact same graphs, but would again, similar to the tools, require inputs to be passed separately, costing much more time and money. Each graph's features and related information are explained in this section.

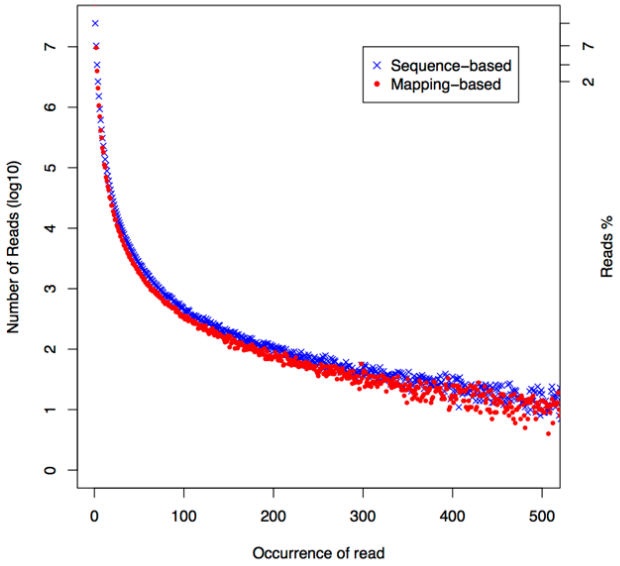


Figure 4: The Occurrence of a read is contrasted with the negative logarithm of the Number of reads to provide the Duplication Rate of that read. Useful for determining quality of data with regard to repetitions.

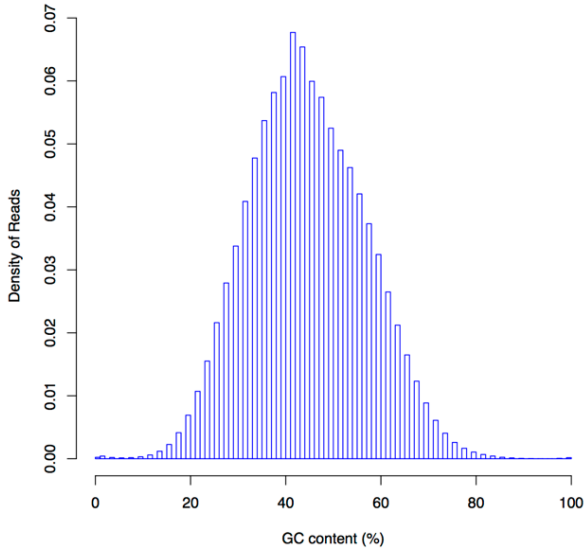
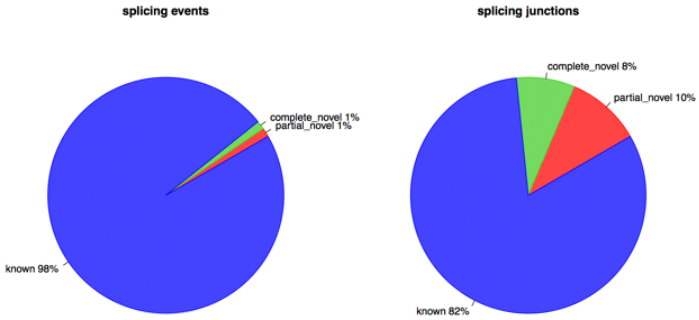


Figure 5: GC Content shows how much Guanine/Cytosine content is contained in the read. The Density of Reads is related to the clusters of reads that are analyzed.

Figure 6: Two pie charts displaying the novel, partially novel, and known events and junctions. These can be used to analyze gene mutations and deviations.



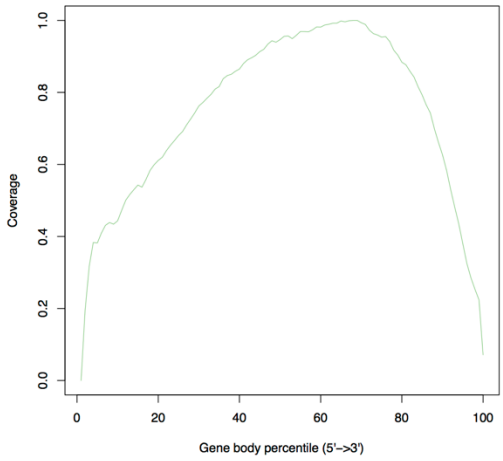


Figure 7: Generated after the RNA Sequencing coverage over the inputted sample is calculated; Helps identify 5' and 3' biases.

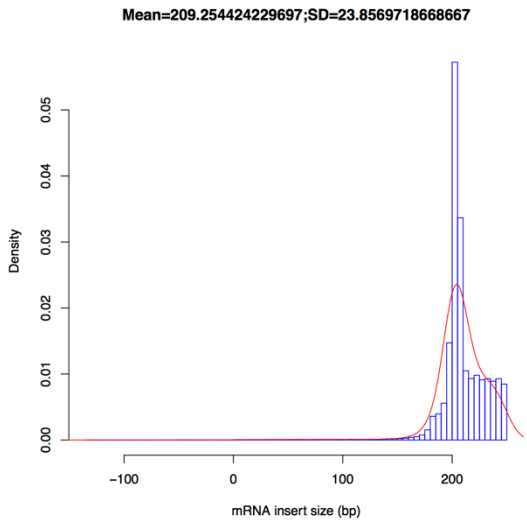


Figure 8: Plot of the inner distance (or insert size) of size between two paired RNA inputs. This data is usually indicative of the uniformity of the reads.

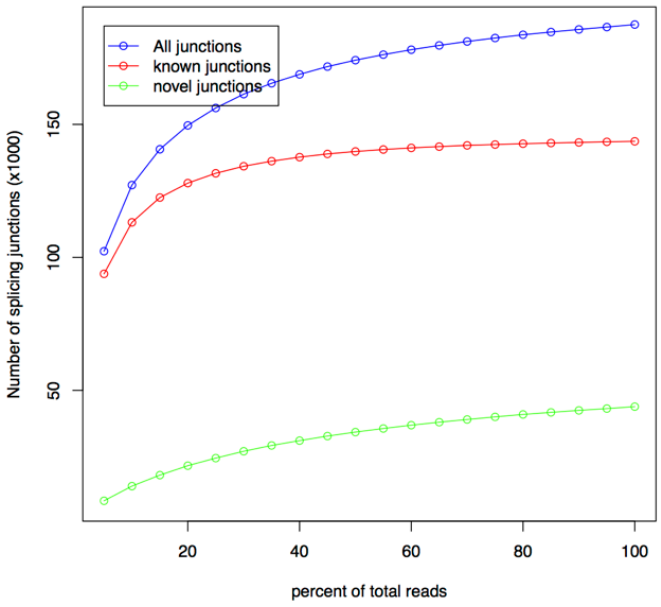


Figure 9: Similar to the pie charts, this graph is generated to analyze whether known junctions are saturated to continue alternative splicing analysis.

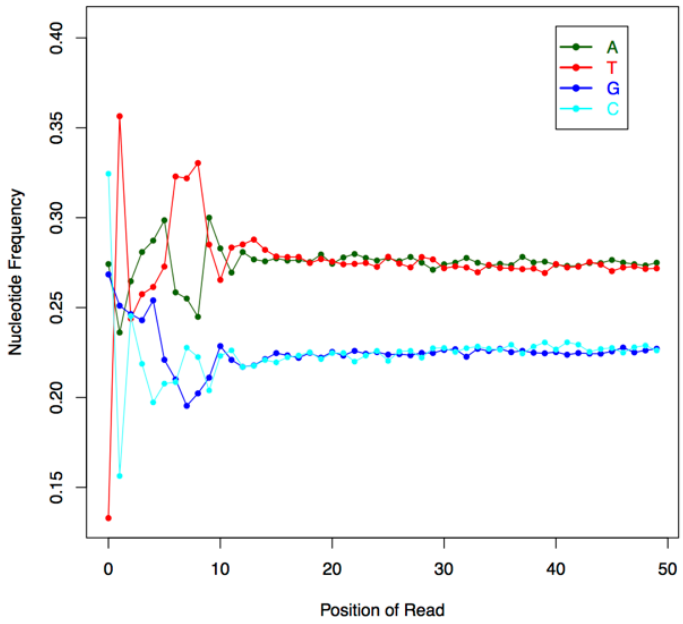


Figure 10: A basic point graph showing the nucleotide frequencies of Adenine, Thymine, Cytosine, and Guanine based on the current position or index of the inputted sample.

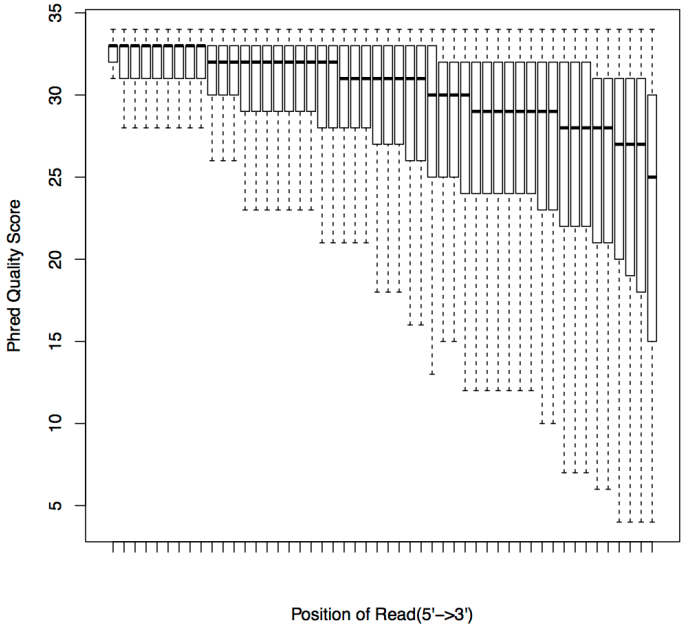


Figure 11: A boxplot of Phred quality score, which is used to determine the quality of automated sequencing procedures.

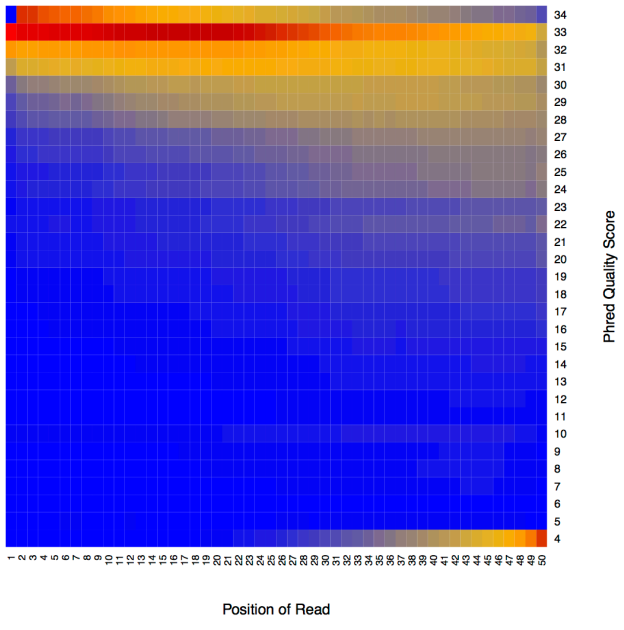


Figure 12: A colored heat map generated of Phred quality score.

All of these graphs facilitate the time consuming error checking process of gene sequencing research, as they provide important quality data in a visual, consolidated manner (Kalari et al. 2014).

5.2 Comparison to Other Methods

5.2.1 Disadvantages

As a pipeline that will be deployed in hospitals, we could not include untested tools, such as STAR and Kallisto, although they are faster. These alternatives are replacements for Mapsplice (Dobin et al. 2013) and RSEM (Łabaj et al.), respectively. STAR and Kallisto are both inherently faster at crunching through data, but are not tested thoroughly enough in the field to be incorporated into a large-scale project such as the RNA-seq pipeline.

5.2.2 Advantages

Some of the key advantages are more related to convenience for the bioinformatician using the software. As mentioned before, the pipeline is ultra portable, able to be downloaded on all operating systems through Docker, which standardizes laboratory procedures. Due to the simple input-output nature of the pipeline, all operations can be initialized faster and in an automatic manner as opposed to being executed by the user one after another. The pipeline also requires minimal dependencies to run, so processing power and storage space are used efficiently. Finally, only one input is needed to start the pipeline, as the features of every tool are mounted onto one accessible cluster for the user to operate with and input files are modified according to the next step in the process.

6 Conclusion

6.1 Impact

The gene sequencing pipeline is not only a more streamlined, user-friendly way to crunch through genetic data, but a rare entry into the toolset of scientists using dated methods match treatments to patients. The portability, lightweight quality, and parallel structure of the pipeline permits a significant increase in accuracy in choosing correct treatments. With the TOIL management system to run multiple processes in parallel and Docker to containerize each element, this pipeline utilizes an extremely efficient structure.

In the real world, this pipeline can be used to arrive from patient data to a diagnosis in a matter of days. Hospitals can send of patient data in a secure and anonymous manner, and receive data files that can be analyzed by doctors to quickly reach a treatment plan. Even more relevant is the application to third world countries with little technology in hospitals; almost no scientific expertise is necessary to pass data through the pipeline, eliminating the need for highly qualified personnel. Doctors in these countries will be experienced enough to look over the supplied data and decide on the next steps for a patient.

6.2 Relevance

Through our pipeline, cancer patients around the world can receive their diagnoses in a much shorter time frame than currently possible. Thus, instead of receiving misdiagnoses, patients are able to rapidly begin personalized treatments. These advancements are achieved through the ingenuity of the structure, the portability of the software, and the scalability of the entire pipeline.

6.3 Future Work

In the future, we hope to continue our research on this novel pipeline structure and develop other methods that coincide with the sequencing of the human genome in order to improve precision treatments for various cancers. We wish to delve deeper into the nuances of each of the intricate pipeline processes and analyze efficiencies and misjudgments in the methods used. Possible future works may include: implementing more advanced tools such as STAR or Kallisto and applying the gene sequencing pipeline to other highly aggressive diseases found in children. In a more general sense, the pipeline structure is one that can be reproduced and improved for projects not related to gene sequencing, or even bioinformatics. The speed and precision to which the pipeline operates at are attributable to the novel structure and management system created for this project.

References

Dobin, A., Davis, C. A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., ... & Gingeras, T. R. (2013). STAR: ultrafast universal RNA-seq aligner. Bioinformatics, 29(1), 15-21.

Felter, Wes, Alexandre Ferriera, Ram Rajamony, and Juan Rubio. "An Updated Performance Comparison of Virtual Machines and Linux Containers." IBM Research Report (2014): n. pag. Cs.nyu.edu. IBM.

Gatta, Gemma, Riccardo Cappoaccia, Michel P. Coleman, Lynn A. Gloeckler, and Franco Berrino. "Result Filters." National Center for Biotechnology Information. U.S. National Library of Medicine, n.d.

Goatley, Hunter. "Info-ZIP's UnZip." Info-ZIP's UnZip. UnZip, 10 Sept. 2009.

Hoadley, K. A., C. Yau, D. M. Wolf, A. D. Cherniack, D. Tamborero et al., 2014 Multiplatform analysis of 12 cancer types reveals molecular classification within and across tissues of origin. Cell 158: 929–44.

Kalari, K. R., Nair, A. A., Bhavsar, J. D., O'Brien, D. R., Davila, J. I., Bockol, M. A., ... & Kocher, J. P. A. (2014). MAP-RSeq: mayo analysis pipeline for RNA sequencing. BMC bioinformatics, 15(1), 224.

Łabaj, P. P., & Kreil, D. P. Sensitivity, specificity and reproducibility of RNA-Seq differential expression calls.

Li, Bo, and Colin N. Dewey. "RSEM: Accurate Transcript Quantification from RNA-Seq Data with or without a Reference Genome." BMC Bioinformatics 12.1 (2011): 323. Print.

Li, Heng, Handsaker, Bob, Wysoker, Alec, Fennell, Tim, Ruan, Jue, Homer, Nils, Marth, Gabor, Abecasis, Goncalo, Durbin, Richard & 1000 Genome Project Data Processing Subgroup (2009). The Sequence Alignment/Map format and SAMtools. Bioinformatics, 25, 2078-2079.

Mardis, Elaine R., and Richard K. Wilson. "Cancer Genome Sequencing: A Review." Human Molecular Genetics 18.R2 (2009): R163–R168. PMC.

Matthes-Martin, Susanne, Ulrike Pötschger, Ronald Barr, Manuel Martin, Heidrun Boztug, Thomas Klingebiel, Andishe Attarbaschi, Werner Eibler, and Georg Mann. "Costs and Cost-Effectiveness of Allogeneic Stem Cell Transplantation in Children Are Predictable." Biology of Blood and Marrow Transplantation 18.10 (2012): 1533-539.

Niemenmaa, Matti, Kallio, Aleksi, Schumacher, André, Klemelä, Petri, Korpelainen, Eija & Heljanko, Keijo (2012). Hadoop-BAM: directly manipulating next generation sequencing data in the cloud. Bioinformatics, 28, 876-877.

Parkhomchuk, Dmitri, Borodina, Tatiana, Amstislavskiy, Vyacheslav, Banaru, Maria, Hallen, Linda, Krobitsch, Sylvia, Lehrach, Hans & Soldatov, Alexey (2009). Transcriptome analysis by strand-specific sequencing of complementary DNA. Nucleic Acids Research, 37, e123.

Quinlan, Aaron R. & Hall, Ira M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics, 26, 841-842.

Russell, H. V., J. Panchal, H. Vonville, L. Franzini and J. M. Swint, 2013 Economic evaluation of pediatric cancer treatment: a systematic literature review. Pediatrics 131: e273–87.

Rybicki, J.; von St Vieth, B., "DARIAH meta hosting: Sharing software in a distributed infrastructure," in Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on , vol., no., pp.217-222, 25-29 May 2015.

Stenberg, Daniel. "Curl." CURL. CURL, n.d.

Tirode, F., D. Surdez, X. Ma, M. Parker, M. C. Le Deley et al., 2014 Genomic landscape of Ewing sarcoma defines an aggressive subtype with co-association of STAG2 and TP53 mutations. Cancer Discov 4: 1342–53.

Wang, Kai, Singh, Darshan, Zeng, Zheng, Coleman, Stephen J., Huang, Yan, Savich, Gleb L., He, Xiaping, Mieczkowski, Piotr, Grimm, Sara A., Perou, Charles M., MacLeod, James N., Chiang, Derek Y., Prins, Jan F. & Liu, Jinze (2010). MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery. Nucleic Acids Research, 38, e178.

Wang, Zhong. "RNA-Seq: A Revolutionary Tool for Transcriptomics." Nature.com. Nature Publishing Group, Jan. 2009.