Comparative analysis of Gene Finding tools when applied to *Trichoderma* genomes

A thesis submitted to the
College of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of Master of Science
in the Applied Computing of Computer Science
University of Saskatchewan
Saskatoon

By
Connor Burbridge, Dave Schneider, Tony Kusalik

Permission to Use

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Disclaimer

Reference in this thesis to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement, recommendation, or favoring by the University of Saskatchewan. The views and opinions of the author expressed herein do not state or reflect those of the University of Saskatchewan, and shall not be used for advertising or product endorsement purposes.

Requests for permission to copy or to make other uses of materials in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building, 110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan S7N 5C9 Canada

OR

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9 Canada

Abstract

placeholder

Acknowledgements

I would like to acknoweldge both Dr. Kusalik and Dave Schneider for their excellent support and mentorship during both COVID and my MSc. project. I would also like to thank Brendan Ashby and Dr. Leon Kochian for providing both data and additional financial support.

Contents

List of Tables

List of Figures

List of Abbreviations

DNA	Deoxyribonucleic acid
RNA	Ribonucleic acid
CDS	Coding sequence
Mb	Megabases
Kb	Kilobase
WGS	Whole Genome Shotgun (sequencing)
NGS	Next generation sequencing
PCR	Polymerase chain reaction
GMO	Genetically modified organism
CPU	Central processing unit
GC	Guanine cytosine
HMM	Hidden Markov model
UTR	Untranslated region
GFF	General feature format
BUSCO	Benchmarking Universal Single-Copy Orthologs
RSMI	Root, Soil and Microbial Interactions
MITE	Minature inverted-repeat transposable element
TIR	Terminal inverted repeat
TDR	Terminal direct repeat
maybe	Should I include tools like BLAST and resources like NCBI?

1 Introduction

The study of organisms in Biology is a highly complex complex process involving many disciplines. To better understand how these organisms function, we must braker down the problem into different sub-problems. One important sub-problem in biology is the understanding of the molecular tools and processes used by cells to function in normal and abnormal environmental scenarios or stress conditions. These conditions may include disease and environmental stress for example. However, previous work in biology has shown that the underlying backbone of information, known as the genome, can vary widely between different organisms in many aspects, such as overall structure, length, ploidy, methylation, and overall gene content. All of these aspects can affect the survival of an organism in any given environment, some of which may be interesting to researchers. As an example, one of the most popular and extensively studied diseases is cancer. Through the study of human genomes, researchers identified a key gene, named TP53, involved in the suppression of tumours. Functional mutations affecting this gene can result in increased risk of cancer. Understanding how and why TP53 suppresses tumours can provide insight into future cancer prevention methods and treatments. These genes can then be mapped to the genome of the organism to identify its location.

This general workflow can be applied to features of interest from organisms in all branches of life. To facilitate this process of identifying genes from a genome, the process of genome annotation was developed. In this case, instead of identifying a gene of interest and then mapping it back to the genome, gene finders identify potential genes from a reference genome before truly knowing their function or if the candidate is truly untilized by the organisms. This set of potential genes acts as a reference for future research. However, to generate a reliable set of possibe genes, a gene finder must be supplied with a suitable high quality genome. In many cases, researchers may be studying a specific strain or variety of organism that differs from the reference assembly and annotation available for the organism of interest. Rather than using the reference assembly for analysis, it may be beneficial to generate a new assembly for the unique variety or strain, which must then be passed through a gene finding tool. With the variety of gene finding tools and approaches, the choice of an appropriate tool can affect the resulting gene set. This problem raises a question. Do the results from gene finding tools differ? And more importantly, how should one compare results from gene finding tools when there is no reference annotation for a specifc variety in question? Most genome annotation tools benchmark their performance in comparison to an existing reference annotation, which is usually considered to be of suitably high quality. This is not possible in the case of unique assemblies, and so the devleopment of a comparative methodology is in order.

2 Background

2.1 Genomics

Genomics is a wide area of study focusing on the genomes of organisms from all varieties of life. A genome is a sequence of characters that contains the fundamental set of 'rules' used to create what we know as life. One can think of a genome as a set of instructions that our cells use in order to complete the tasks that make us function. A genome is comprised of tightly bundled sequences of DNA, which are stored in the nucleus of cells. These bundles of DNA contain sections known as genes, which can be thought of as the tools described by the set of instructions. These tools carry out a vast number of processes ranging from no known function at all to genes that are key in protecting against diseases cancer. Genomes can vary widely in size, ranging from small bacterial genomes of roughly 4 Mb up to approximately 149000 Mb. Piecing together genomes provides numerous opportunities to understand other 'omics' within cells, such as proteomics, metabolomics, transcriptomics and epigenomics.

2.1.1 Sequencing

Sequencing data is a pivotal form of data used in nearly all applications of Bioinformatics. To understand the processes used by organisms for day to day survival or in unique circumstances, we must have an initial set of data points to work with. These sequences, referred to as reads after sequencing, are the foundation for solving problems ranging from taxonomical classification to the understanding or complex biological functions like signaling pathways. Reads may come in a variety of forms and formats depending on the desired application.

2.2 Whole Genome Shotgun Sequencing

Whole Genome Shotgun sequencing (WGS), is a method to produce a large number of genomic sequences from a sample of interest. This is form of sequencing is quite common as it is has a wide variety of applications in research [1]. WGS involves slicing up genomic DNA into smaller segments. These small segments are then processed further resulting in a set of physical molecules that can be supplied to a compatible sequencing platform of which there are a variety. Modern sequencing platforms are comprised of next generation sequencing (NGS) and 3rd generation sequencing approaches.

2.3 Next Generation Sequencing - Illumina

Illumina sequencing is one of the most popular NGS platforms currently available. Illumina sequencing produces a very large number of high quality short reads, typically between 75 and 250 base pairs in length. Sequencing libraries can be prepared to produce reads solely from one end of a sequence fragment (single-end) or both ends (paired-end). Advantages of paired end sequences are the additional context provided by the paired sequence on the opposite end of the fragment. This context is leveraged by read processing tools to identify features such as repetitive regions and genomic rearrangments, which can be significant in downstream analyses. Illumina sequence librairies are generated by first fragmenting the DNA samples, amplifying them via PCR, ligating adapters that allow the sequence to bind to the sequencing plate, and finally identifying each fragment's sequence of nucleotides using fluorescently-labeled nucleotides that bind to the fragments [5].

2.4 3rd Generation Sequencing - Nanopore

Nanopore sequencing data is relatively recent approach to sequencing projects. While Illumina reads are considered to be short, Nanopore reads are much larger, ranging from 10Kb to 300Kb depending on the approach used. Long reads are beneficial due to their ability to bridge the gaps between difficult to assemble regions when performing sequence assembly. An example of a difficult to assemble region would be a region with a high repeat content, where a large number of small repeats may be collapsed during the assembly process, resulting in an assembly that does not represent the true nature of the sequence being studied [7]. Whlie Nanopore was previously known for having lower quality base calls when compared to Illumina, that is no longer the case at this time. Nanopore sequencing works by passing long seqments of genetic sequence through a membrane bound protein and measuring changes in electrical current, which is characteristic of the nucleotide at a given position.

2.5 Trichoderma

Crop resistance to environmental stressors is a necessity for crop health and overall crop yields. Current popular methods for crop protection involve the use of pesticides and genetically modified organisms, which can be expensive and potentially politically dividing in the case of GMOs[13]. In addition, crops suffer when soils are not sufficient for crop growth and health. Soil insufficiencies can result in drought stress as well as nutrient stress, leading to poor overall yields.

Trichoderma is a fungi that can both communicate with and colonize the roots of plants in a non-toxic, non-lethal, opportunistic symbiotic relationship[16]. Many strains of *Trichoderma* have been shown to provide resistance to pathogenic bacteria and other fungi in soils through the use of polyketides, non-ribosomal peptide

synthetases and other antibiotic products[16]. Recently, two strains of *Trichoderma* have been identified in the prairie regions of Alberta and Saskatchewan. These two strains, named Tsth20 and DC1, have been found to have beneficial properties when used as an inoculant for plants in the soils mentioned before. In addition to these beneficial properties, the two strains mentioned previously provide even further protection for plants in dry, salty soils and one strain also has potential for use as a bioremediation tool in soils contaminated with hydrocarbon content. Bioremediation and resistance to drought tolerance has also been investigated in other strains of *Trichoderma* as well[11]. However, little is known about the mechanisms at work in these strains, so DC1 and Tsth20 were sequenced by the Global Institute for Food Security (no publication yet) in an initial attempt to better understand the details of these genomes. While this research does not directly identify genomic elements related to the secretome of these genomes, it may serve as a foundation for future research of *Trichoderma*.

2.6 Genome Assembly

Sequence assembly has been a long-standing problem in the field of bioinformatics[8]. Determining the correct order and combination of smaller subsequences into an accurate complete sequence assembly is computationally difficult in terms of compute resources such as memory, CPU cycles and storage required for input sequences[8]. In addition to these difficulties, there can be other issues encountered during asssembly due to the nature of the data or genomes themselves, such as low quality base calls for long read data, which is not necessarily the case today, or the inherent content of genomes themselves using repetitive regions as an example. Insufficient data used in an assembly may result in short, fragmented assemblies, depending on the size of the genomes, while sequence data that is not long enough can fail to fully capture repetitive regions in an assembly. To solve this problem, a wide range of assembly tools have been developed with their own unique approaches to the genome assembly problem, so it is important to use an appropriate assembler for the task at hand, and also important to evaluate the assembly thoroughly.

Genome assembly tools generally approach the assembly problem using a graph-based approach. The most common graph-based approach is the de Bruijn graph assembly [3]. A graph in this context, is set of nodes (k-mers from sequences) connected by edges (overlaps between k-mers). Traversing through this graph results in longer subsequences that ultimately result in a set of consensus sequences and final assembly. In the early years of long read sequence data, sequencing platforms encountered difficulties producing consistently high scores for base calls when seuqncing. To combat this, some assembly workflows may also include a polishing or correction step once the initial assembly is completed in which high quality short read sequences are supplied as supplemental information to correct low quality base calls in the assembly. These low quality base calls are typically not present in modern long read sequencing approaches as the methodology and quality of calls have improved drastically. While the polishing step is arguably unnecessary in modern assemblies, the polishing programs remain available should researchers be interested in applying additional reads for

polishing.

One approach to aid in the previously mentioned issue of assembly correctness is to use a combination of long and short reads in what is known as a hybrid assembly. Combining both highly accurate short reads with deep coverage along with less accurate but much longer reads can produce high quality genome assemblies that capture long repetitive regions. Hybrid assembly approaches have been shown to produce high quality assemblies in a wide variety of organisms as the combine long read data with short data to produce assemblies that properly represent long repetetive regions with additionaly high quality Illumina sequences for correction. Once assembled, the sequences must also be evaluated with measures such as N50, L50, coverage, average contig length and total assembled length to ensure that the genomes are well assembled, at least based on these metrics[8]. Following appropriate assembly protocols is essential to the further success of a project as downstream processing such as annotation depends on a high-quality assembly.

2.7 Identification of AT-rich Genomic regions

One important aspect of interest when assembling any form of sequence is GC content or percent GC of the assembled sequence. Large regions of anomolous GC content may be of interest to researchers as they may contain repetitive regions and unique features responsible for traits specific to the organism in question.

2.8 Repeat Identification and Masking

Repeat identification within assembled genomes is a problem that needs to be considered during the genome annotation process. Regions with long repeats can have a significant impact on genome assembly as well as gene finding due to the limitation of short reads used in some assemblies[14]. Short reads may be unable to bridge or cover entire repeat regions within a genome, so it is important to consider the use of long reads from technologies such as Nanopore or PacBio to provide a complete picture of these regions when pursuing a new genome assembly project. It is also possible for repetitive regions to contain genes as well, making for an interesting investigation in regards to *Trichoderma*, as fungal genomes have been shown to contain many repeat regions with a high concentration of A and T nucleotides[15]. Once these repetitive regions have been identified, the genome could be masked to exlude these regions in downstream processing if desired, as these regions may be poorly assembled and may result in found genes that do not truly exist in those regions. However, this may not be as common today, as repetetive regions have been shown to contain genes as well[12]. This may affect the gene finding process described later and may be an interesting topic to look into considering the large number of available gene finding programs.

2.9 Centromere Identification

A centromere is a region of a chromosome that is crucial for the proper cell division. These regions are the main anchor for microtubules, which are a cellular structures used that attach to centromeres to separate chromosomes during both mitosis and meiosis. Centromeres are critical to the survival of an organism, with malfunctions in the process of cell division usually resulting in potential disease and fatal outcomes[10]. Centromeric sequences can be comprised of several different genetic components, with repetititve regions being the most prevalent in the forms of satellite DNA and transposable elements. In addition to centromeric regions, there are flanking pericentric regions with their own properties, including potential candidates for small-interfering RNAs[10]. Identification and consideration of centromeric regions may prove useful when comparing the outputs of gene finding tools, as the underlying properties and structure of the genetic sequence differ in comparison to typical coding regions of DNA.

2.10 Gene Finding Methods

Gene finding (or gene annotation) has been a long standing computational problem in bioinformatics, which concerns itself with identifying potential genes within assemblies based on patterns or pre-existing experimental evidence evidence considered by the gene finding program. This process is critical for unraveling and understanding the complex processes occurring in all forms of life with applications in medical science, agriculture, biomanufacturing, environmental studies and many others. In a general sense, gene finding programs operate by searching for patters or indicators showing that a gene of feature may be present. The most basic indicators being start and stop codons, with introns and exons in between should the sequence match the applied model. The results produced by gene finding tools can vary considerably for a number of reasons, including quality of the assembly, the intrinsic model used by the gene finder, filtering criteria, and even the nature of the organism and assembly itself. Given the broad applications, choice of gene finding tools, and the variability of assemblies being considered, it is important that we gain a deeper understanding of these tools prior to putting them to use.

There are two common methods for gene finding, those methods being *ab initio* methods, where programs search for patterns and gene structures, and similarity or evidence-based searches, which use prior information such as RNAseq data, expressed sequence tags and expressd protein sequences to identify genes within a new genome[4]. Complicating the process more is the introduction of introns and alternative splicing in eukaryotes, making it possible for one gene to have several possible transcripts at the same locus. An example of an *ab initio* method would be GeneMark-ES[6], while an evidence based tool would be Braker2[2]. *Ab initio* gene finders typically predict genes using a Hidden Markov Model (HMM)[4]. These predictions are based on 'signals' or features associated with a gene, such as the usual start, stop, exon and intron portions of a gene as well as upstream promoter sequences and more. In this case, these signals would be considered states

in the terminology associated with HMMs. Gene finders wish to predict these states based on observations, or sequences presented to the model. HMMs in gene finding tools are trained beforehand and then applied to a sequence. This means that a gene finding program may not be trained in the context of any assembly provided to it, and thus may miss genes that are unique to the assembly in question. On the other hand, while still relying on HMMs for a 'base' set of predictions, evidence-based gene finding tools leverage new evidence that may be outside the scope of the pre-existing model[**?**]. As an example, an evidence-based model would be useful in a situation where you are interested in annotating a new assembly for a non-model organism. The addition of experimental data provides context specific to your assembly of interest while still retaining the predictions from existing HMM models.

There are also other aspects of gene finding tools that are important to consider. These include features such as whether or not the gene finders find non-coding RNAs, annotation of 5' and 3' UTR regions, and in the case of ab-initio methods, the assumptions made by the underlying models used for gene finding. These features and others can influence a user's decision on which gene finding tool to consider and will complicate comparative analysis of multiple gene finding tools. (citation needed somewhere in here)

2.11 InterProScan

The outputs from gene finding tools are a set of potential genes that fit the model used by each tool. While they are considered genes, the use of the word gene is used in a very loose sense, in that these genes may or may not be functional or match any existing gene sequences from previous research. Typically, to confirm the 'correctness' of predicted genes, the outputs from a given tool are used in a sequence similarity search against a reference set of genes or a large datasbase comprised of multiple organisms. This approach is straightforward, but can introduce bias from database choice and also allows for vague or loose matches, depending on the parameters used and the interpretation of the results. Another approach is to use InterProScan, which is a tool used for functional annotation of proteins using evidence from a variety of databases [**?**]. The presence of some form of functional domain or annotated structure in a predicted gene sequence is reasonable evidence for the existence of a predicted gene. In addition. This approach also avoids the problems associated with similarity-based approaches.

2.12 File Formats

2.12.1 FASTA

One of the most popular formats for sequences of DNA, RNA and amino acids is the FASTA format. The FASTA format consists of one or more entries containing two or more lines. The first line of an entry is the ID line, which must begin with a greater-than ('¿') character, followed by an ID and any other pertinent inormation for the following sequence. The greater-than character is the indicator that a new sequence has

```
>rna1
ATCGTAGTGCTGTGATCGTAGTCGGATTGGACATGATCGATTCGAT
TGATTGATCGATTGCTAGAAATCGATCGGCTCGTATATCGATCGTA
ATGTCGTTAGTCGAT
>dna2
ATTAGTCGATGCTAGCTGATGGTTAGCTAGTTCGATTCGGTATAGATCTAGGCTTAGAGATATCGCGCTAGCTAGCTAGC
```

Figure 2.1: Example of two FASTA sequence entries. One example with sequence characters split across multiple lines, and one showing all sequence characters on the same line.

```
@M00833:808:000000000-CJYGN:1:2105:15607:1332 1:N:0:1
CAACTGACGTAGGTGGGACATTCTCCAACGATGATCCTGCGAGGAATGCGATGGAGGGGCGCAGATGCGGGTTTTGGTTCTAGATG
CCCCTGCAGCGGTTCGCATGTGGCTTTGCCTTTTTTTCTTTTTTTTTTTTTTTTGTTTTTCTGTTTTTTTAGATGTTCGGAACAGC\
CTCTTTTTGGACTTTTTTTTTTGGAAACTCGCTGCTGGAATGCTGGGTTTTGGTCCCTTTTAGGTTTTGGCTTACTGTT
+
AABCBFFFBAAFGFGGGGGGGGHHHHHGFGGHGFHHHHGHGGGGGBFGHGGHGEHHGGGGGGGGGGHHGGGEFGGHGHHHHHHGGH
HHHGHGHHGHGGGCFGGGGGFHHHHHHHHHHHHGHGGGHHHHHGGGGGFGC-;-;BFFE/00:FFF.-D.B000;FBA-..-//:\
9//;BFFF=-;0;FFFFFFFF;-::B9/00.-...0;0.;009000./.9;.-:0F.F00;0:0009..-00:0;00;0
```

Figure 2.2: Example of the four lines in a FASTQ entry.

begun. The following line(s) contain the actual sequenced nucleotides or amino acids, which can be contained on one line or split across many lines. An example of multiple FASTA entries are shown in figure 2.1.

2.12.2 FASTQ

Another popular sequencing format is the FASTQ format. This format is very similar to the FASTA format but with the addition of two more lines per sequence entry and a change to the character indicating the beginning of a new sequence entry. An example of a FASTQ entry is shown in figure 2.2. In FASTQ formatted entries, the greater-than ('¿') character is swapped with the at ('@') character. The IDs for the sequence also follow a specific format, which provide information about the sequencing run and flowcell that the read was sequenced on. This information can then be traced back to the sequencing experiment in the case that there were errors or anomalies in the output from the experiment. Following the ID is the string of base calls. The third line in a FASTQ entry is a plus ('+') character, which indicates that the sequences character line has finished. Following the plus ('+') character is another sequence of characters, this time indicatintg the quality of basecall for the corresponding nucleotide base calls in the second line. The quality information included in FASTQ files are used to assess the quality of a sequencing run and extensively used in downstream processing steps, most notably in alignments.

2.12.3 General Feature Format - GFF

General feature format (GFF) is a popular format for storing information about features relative to a position on an genetic sequence, and comprises a large portion of annotation results from this work. These features can be whatever the user desires, as long as the feature entry follows the required GFF guidelines. Relative to a reference sequence, each GFF entry contains the following tab-delimited columns: sequence ID, source, feature type, start position, end position, score, strand, phase, and a semi-colon delimited list of attributes. GFF files are widely supported accorss bioinformatics tools, making them highly versatile while

```
ctg000000    AUGUSTUS    gene  10842 11309 .     -     .     ID=g4;
ctg000000    AUGUSTUS    mRNA  10842 11309 0.6   -     .     ID=g4.t1;Parent=g4;
ctg000000    AUGUSTUS    stop_codon  10842 10844 .     -     0     ID=g4.t1.stop1;Parent=g4.t1;
ctg000000    AUGUSTUS    CDS   10842 11309 0.6   -     0     ID=g4.t1.CDS1;Parent=g4.t1;
ctg000000    AUGUSTUS    exon  10842 11309 .     -     .     ID=g4.t1.exon1;Parent=g4.t1;
ctg000000    AUGUSTUS    start_codon  11307 11309 .     -     0     ID=g4.t1.start1;Parent=g4.t1;
```

Figure 2.3: An example of GFF entries for a sinlge gene.

also remaining relatively simple in nature but also allowing for storage of more complicated items via the attributes column. One significatn useage of GFF files is in visualization of features against the reference sequence from which they were derived. Most genome viewers (or browsers) support GFF files as input, allowing intuitive visualization of many features when overlayed on a reference sequence. An example of a GFF entry can be seen in figure 2.3.

Background for BUSCO (from research questions) As more and more genomes are assembled for new organisms, a tool was developed to evaulate assemblies and subsequent annotations from the pserpective of gene orthology. As genomes diverge evolutionarily, it is expected that some genes will be conserved. The BUSCO (Benchmarking Universal Single-Copy Orthologs) tool and datasets were developed to assess completeness of an annotation in comparison to evolutionarily conserved genes.

3 Research Questions

3.1 Research Questions

With an ever-increasing number of gene prediction tools available to users, it is important to assess and understand their behaviour and performance in the context, particularly in the context of new genome assemblies of lesser studied organisms, where a reference prediction set may not be available. The main purpose of this research is to evaluate and compare gene finding tools in the context of *Trichoderma* assemblies where a gold standard set of gene predictions does not exist. To assess behaviour and performance in these contexts, we have defined X problems to profile the selected gene finding tools. In addition to applying selected gene finding tools to novel *Trichoderma* isolates, Tsht20 and DC1, we also applied selected gene finding tools to existing *Trichoderma* assemblies from the National Center for Biotechnology Information (NCBI).

3.2 *Trichoderma* Assembly Results

Since gene finding tools operate on an assembled genomic sequence, it must follow that the results will be influenced by the supplied assembly. Before applying gene finders to the new assemblies, we should first investigate the new assemblies by generating general assembly metrics for the new assemblies to contrast and compare with existing assemblies. We ask: **how do assemblies of DC1 and Tsth20 compare to existing *Trichoderma* assemblies?** With these isolates being from the *Trichoderma* family, we expect assembly metrics to be similar in nature to existing assemblies from NCBI, but do they?

3.3 Profiling of Gene Finding Tools

Different gene finding tools may predict different types of features associated with gene structures. The question arises: **which (if any) gene finders predict additional features outside of the standard gene model?** Additional features in this case include promoter sequences, transcription binding sites, activating sequences and other upstream or downstream sequences. In addition, different gene finding tools employ differing programming languages and algorithms which raises several questions. **How are these gene finding tools implemented? Is the software straightforward to install? Are the tools user-friendly? What is the processing time and memory consumption of different gene finding**

tools in the context of *Trichoderma*?

3.4 Number of Features Predicted

Do gene finders predict similar numbers of features in the context of *Trichoderma* genomes? One common method for evaluating gene finding tools is by looking at the number of features predicted by each tool. These features make an obvious point of comparison for selected gene finding tools. The term 'feature' here is somewhat ambiguous, referring to many possible categories of genomic feature. For each gene finding tool, we compare the counts for predicted genes, transcripts, and coding sequences.

3.5 Lengths of Predicted Genes

Genome assemblies can contain a wide range of gene lengths. For some users, genes of a specific length may be a key point of interest, so the ability of a gene prediction tool to capture the broad range of possible gene lengths is another important metric for comparison. Thus we ask the question: **do different gene finders predict genes of similar lengths in *Trichoderma*?**

3.6 Performance in Regions of Anomalous Sequence Content

One of the inspirations for this research is the unique composition of genomic sequence in *Trichoderma*. Results from the assembly process show that GC content in *Trichoderma* strains is abnormal throughout most assemblies. These regions of assemblies present an interesting opportunity to assess gene finding performance in regions of anomalous GC content. The question follows: **do gene finders behave differently in regions of anomalous sequence content?**

3.7 BUSCO Completeness

The use of existing benchmarks for gene finding performance is useful when assessing performance of gene finding tools, particularly in the case of genes that should be evolutionarily conserved. **Do gene finders predict conserved single-copy orthologs expected in fungal genomes?**

3.8 Identifying Regions of Agreement and Disagreement

With predicted genes from several tools available, the question we would like to ask is whether or not the gene finders agree with one another for any given prediction. To answer this question, we will identify 'regions' of overlapping predictions. A region can be defined as a start and stop position of a set of individual or overlapping features from one or more gene finding tools and external sources. With regions identified,

we can determine agreement, or more importantly, disagreement in predictions between gene finding tools from which we can ask: **do gene finders agree on their predictions? If no, to what extent do they disagree? Are there genomic regions where agreement or disagreement are more prevalent?**

3.9 Validation of Predicted Genes via InterProScan

In an effort to validate, or at least provide supporting evidence for any given gene prediction we will apply InterProScan to coding sequences predicted by each of the gene finders to identify features associated with protein function. Genes will be considered as 'valid' if the gene's protein sequence contains binding sites, motifs, or other fucntional characteristics of proteins. Using the results from InterProScan, we ask the question: **in the context of *Trichoderma*, do proteins predicted by gene finders contain functional signatures?**

3.10 Identification of a Core *Trichoderma* Gene Set.

Using results from identification of regions in section 3.8 we will identify a shared or 'consensus' set of predictions for each *Trichoderma* assembly. **Can we identify a set of consensus predictions from gene finding results? What are the properties of the resulting set?**

3.11 Selection of a Gene Finding Tool

With all the results generated, we can provide insight to the question: **which gene finding tool should one choose?**

4 Data and Methodology

4.1 Methodology

4.2 Data and Processing Overview

4.2.1 Data

Comparing the performance and features of gene finding tools, both qualitative and quantitative, in the context of any set of genomes is important for those interested in selecting a specific gene finding tool. To accent(?) the processing for genomes of interest, those being DC1 and Tsht20, we should include other previously assembled *Trichoderma* assemblies. Currently selected genomes include *Trichoderma reesei*, *Trichoderma harzianum*, and *Trichoderma virens*, with *Trichoderma reesei* being the 'reference' in this case, as it is well studied and there are several patents involving it's use a organsim for production of compounds such as antibiotics in industrial applications.

The general methodology for this work is described in figure 1. Each portion of this figure is discussed in detail in this section.

4.3 Assembly and Annotation

In an attempt to produce high quality assemblies of DC1 and Tsth20, We decided on a set of tools named NextDenovo and NextPolish as they have produced excellent assemblies based on previous experience. (should find a citation to confirm this)

(Might be better for discussion or omitted since it is specific to our setup) Initial attempts to run the example dataset resulted in permissions errors due to the management of the storage system being used, which were encountered with other tools in the past. To remedy this, the software installation was copied to RSMI's scratch space on Copernicus. Once the approriate permissions were given to run nextDenovo, the example dataset was run without issue.

Following assembly using nextDenovo, Illumina sequence data from DC1 and Tsth20 was used to polish each respective genome using nextPolish. Default parameters were used from assembly except for modification of the parallel option to reduce processing times.

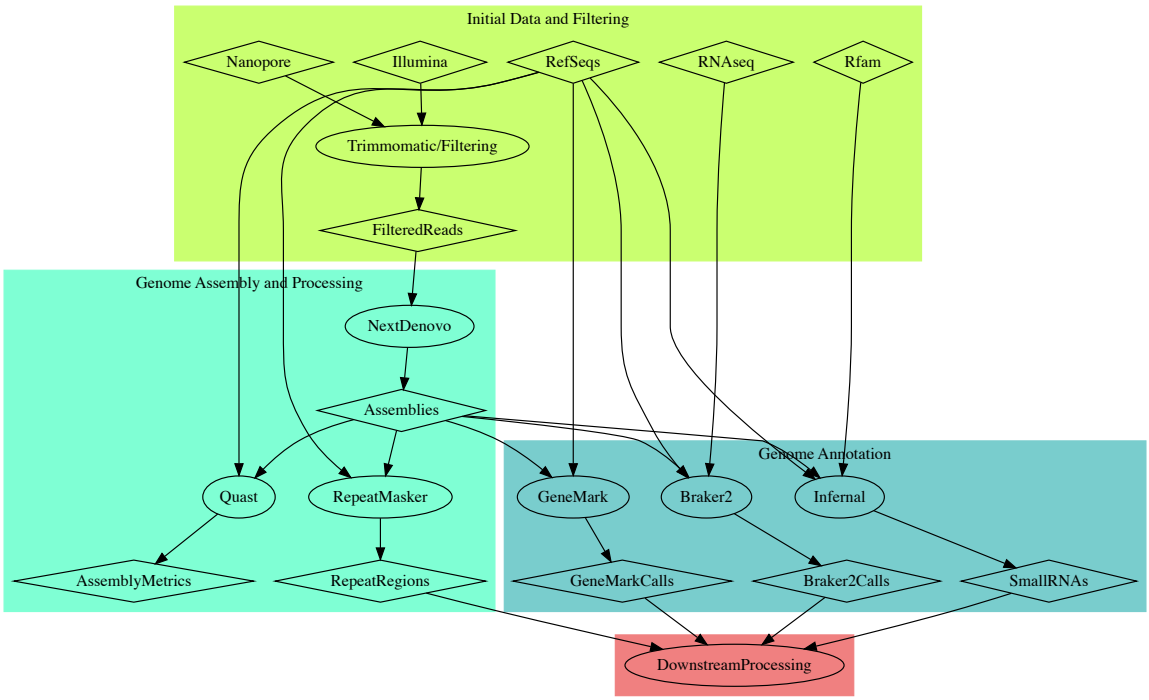


Figure 4.1: A flowchart of the methodology followed for this research. Sections are separated based the general process they are associated with (i.e. input data, assembly, gene finding and downstream analysis).

4.3.1 Repeat Masking

In order to evaluate the performance of gene finding tools in repetitive or low complexity regions in the context of *Trichoderma* genomes, we must first identify said regions in the genomes considered. To do this, the GenericRepeatFinder tool was used, which is a *de novo* repeat detection tool [**?**]. GenerifRepeatFinder detects three different types of repeats, those being MITEs, TDRs and TIRs. Commands used for this program follow the example commands provided on the GitHub page for the GenericRepeatFinder project.

4.3.2 GeneMark-ES

To begin, GeneMark-ES was run as it requires no prior information or alignments in order to run. In this case GeneMark-ES has an option specifically for fungal genomes, which was used in this case. Apart from the fungal option, the only additional options supplied were for output format of GFF3 and number of cores for reduced processing time.

General command structure for GeneMark-ES:

gmes_petap.pl –ES –fungus –format gff3 –cores 48 –sequence /path/to/sequence

4.3.3 Braker2

As mentioned previously, *Trichoderma reesei* was selected as the ŕeferenceǵenome for this work. With this in mind, several short read archives (SRAs) from *T. reesei* were selected for Augustus training. Following Augustus training, the model for *T reesei* was applied to all genomes considered. Settings and procedures from running Braker2 are described below.

The variables that need to be set are AUGUSTUS_CONFIG_PATH and TSEBRA_PATH. Augustus, by defuault, tries to write species information to the location where the software is installed. In this case, we don'thave write permissions to the compute canada software stack hosted byt Research Computing, so the AUGUSTUS_CONFIG_PATH variable must be set in order to create a writeable directory. As long as that path has a directory within it called braker, and a species directory within the braker directory, things should go smoothly. TSEBRA is a set of scripts also made by the creators of Braker and is required to merge results from the various gene prediction tools involved in the Braker2 pipeline. The TSEBRA_PATH simply points to the directory where TSEBRA is located Both Braker2 and TSEBRA can be cloned directly from GitHub (links to come)

4.4 Identification of Overlapping Features and Regions

Feature Identification: To first undertand how gene prediction tools perform in comparison to other gene prediction tools, we must identify features. This identification of features will help us descirbe the similarities, and differences between gene finding tools. A feature, in this context, is any feature stated within a Genomic

Feature Format file (GFF) provided to the program, in which mutliple GFF files can be provided. The definition of a feature, for this application, is an object that contains a contig ID, a start position, an end position and a strand property. In the context of features on different strands, start and stop positions of features are sorted based on left and right positions of the feature in respect to the reference sequence.

Region Identification: In addition to feature creation, we will also identify regions of overlapping features based on the precitions from each gene finding tool. These regions will help identify the agreements, or disagreements, between different gene-finding tools. A region, in this context, is a set of overlapping features, all of which overlap at least one other feature in the region. With each overlap, there will be an overlap type. These types can be defined based on Allen's Interval Calculus (reference), with the exception of features that start beyond the end point of the current region.

======= Example command for braker2:

/scratch/p2irc/p2irc_rsmi/cbe453/masters/software/braker2/BRAKER/scripts/braker.pl –gff3 –threads 60 –TSEBRA_PATH=/scratch/p2irc/p2irc_rsmi/cbe453/masters/software/braker2/tsebra/TSEBRA/bin/ –genome /path/to/sequence –species=TreeseiFungal –fungus –useexisting

BUSCO methodology (from research questions) The BUSCO method was applied using two BUSCO subsets, one generally applicable for fungi, and another targeting an evolutionary branch more closely related to *Trichoderma*.

Stats for length analysis (from research questions) The first statistical tool to be applied is ANOVA (analysis of variance) to compare the mean lengths genes predicted by each gene finding tool with the null hypothesis being that the mean of predicted gene lengths should be the same across all tools considered. In addition to ANOVA, pairwise comparisons of the distributions using a Kolmogorov–Smirnov test is appropriate. The null hypothesis in this test would be that the gene lengths are sample from the same distribution.

Stats for binomial tests (from research questions) To do this, a binomial test will be used, with the null hypothesis being that the number of genes predicted in regions of normal and abnormal GC content should be proportional to the length of normal and abnormal GC content regions in the assembly. For example, if 30 percent of the genome is comprised of anomalous GC content, then we would expect 30 percent of predicted genes to be present in those regions. In addition to anomalous GC content, this test can be applied to repetitive content in assemblies as well.

Stats for regions (from research questions) From these results, Venn diagrams will be generated with Jaccard index calculated for each combination of gene finding tools. The region identification process can also be extended to include features identified by other tools, such as BLAST hits to validated gene models from other organisms and small RNAs. Chi2 goodness of fit tests can then be applied to counts of 'validated' gene predictions or other features with the same null hypothesis that gene finders should predict the same number of features.

4.5 Analysis of Results

After completion of the processing portion of this work, the results must be processed in a useful way, which includes both the biological implications of the gene calls as well as the computational, or gene finding features, of the the selected programs. To better understand how gene finders perform in these two classes, we must define an appropriate plan for analysis of the results produced so far. Currently, downstream analysis plan has been broken down into several sections.

4.5.1 Basic Analysis

Basic analysis of gene finding results is an important part of this research. Total gene, transcript and protein counts will be identified for each genome and gene finding tool combination. Comparing the general outputs of these programs will provide an idea of their performance in different *Trichoderma* genomes In addition to these basic outputs, analysis will also be performed for the following: distribution of gene lengths, intersection of gene calls, smallRNAs and repetetive regions, shared gene content with a close fungal relative. Analysis for these results can be performed through simple shell scripting with grep and other unix tools, although processing through Python might provide results that are easier to reproduce with proper programming. Having one script with several modules that can be rerun at will would be easier to handle than multiple shell scripts. This thinking for processing will be applied to subsequent sections of this as well.

4.5.2 Distribution of Gene Lengths

One important aspect of gene finding tools to consider is the distribution of gene lengths predicted by each individual tool. Certain tools, such as GeneMark are based on pre-defined models, which may limit the length of predicted genes, while tools such as Braker2, which incorporate RNAseq data, may predict a wider distribution of gene lengths depending on the input dataset used. Regardless, the ability of a gene finding tool to predict a wider range of gene lengths can be usefull if users are looking for short or larger genes. To help determine whether or not these tools find shorter genes, or small RNAs, the genomes of interest have been annotated using Infernal along with the Rfam database to identify small RNAs as a ground truth. These annotation results will also be included with results from other annotation processes further down the line. Again, these results can be produced with a Python script. The resulting data could then be used as input to violin plots for each genome and set of tools considered in this analysis process. Violin plots should provide a good visualization of gene lengths as well as the number of genes found with specific lengths. Means could also be compared staatistically for genomes and the mutliple tools considered as well.

Analysis of gene lengths was performed using a Python script. Combined predicted CDS sequences for each predicted gene were used as input for the total gene length. CDS sequences predicted by Braker2, were directly available in the output directories when the program was run. CDS sequences from GeneMark required extraction of the CDS sequences from the genome FASTA files. This process was performed using the

gffread tool from the Cufflinks package. Predicted CDS sequences were loaded into Python using Biopython's SeqIO package. Sequence lengths were then placed in a list and analyzed using a combination of pandas and numpy. A log10 transformation was applied to the sequence lengths as the original distribution was heavily skewed due to long outlier CDS sequences. After transformation, the CDS length distribution appears as a normal distribution, although there are interesting troughs that occur in several of the peaks for several of the genomes considered. These troughs did not appear in a comparable Yeast reference dataset, although the log transformed data still appears to be a normal distribution.

4.5.3 Intersection of Gene Calls, smallRNAs and Repetitive Regions

Annotation of all three features in the title are important in assessing the ability of gene finding tools. Even more important, is the potential for for overlap between gene calls and small RNAS as well as repetitive regions the genomes. As discussed in the previous subsection, the distribution of gene lengths predicted by a gene finding tool can be an important metric for users. Overlapping predicted genes from tools alongside the output from Infernal and the Rfam database may provide insight into whether or not these gene finding tools are able to predict RNAs of very short length. In addition to small RNAs, repetitive regions in *Trichoderma* genomes hold potential for recombination and gene content, although the inherent nature of these repetitive regions (low nucleotide diveristy) suggests that gene content should be low, based on the nucleotides required for start and stop codons. Analysis of these intersections can be perfomed via bedtools or through biopython (I believe). Again, having all processing steps included in one script as separate functions that can be called at whim will make further processing easier if changes need to be made.

4.5.4 Methodology for Indetifying Overlapping Features

To analyze the results from multiple gene-finding tools, we must first define two conceptual topics.

Definition of a Feature: First, a feature, in the context of this research, is any item contained within a Genomic Feature Formated file (GFF). Each feature contains a contig ID, a start position, end position, and a feature ID based on the information from the GFF file. These features will be used in the process of identifying regions, or overlapping features from prediction tools.

Definition of a Region: Secondly, a region is defined as any overlap between features relative to the reference sequence being considered. To identify regions, every feature from each GFF file being considered, is sorted by start position. For clarification, the start position is based on the left most position in the GFF file, regardless of the strand that the feature is predicted on. Once the features have been sorted by left position, the sorted features are iterated over to identify regions, as long as the left position of the next feature is consistent with the left position , within the left and right position, or equal to the right position of the current region. One drawback to this approach is that ideally, identification of overlap types based on Allen's interval algebra would be performed at this point. However, the methodology of initially sorting features based on left position somewhat prevents this process from happening. This implementation requires

further processing of identified regions late on the process, simplifying the all to all comparison that would occur if all features were considered at once.

Identification of overlapping features results in different 'classes' of regions. A few basic cases are outlined in figure 4.2. The simplest of case is the a region where the gene finders agree unanimously on the gene (agreement currently means same start and end point).

4.5.5 Shared Gene Content with Closely Related Organisms

While considering novel gene calls can be useful, comparing those calls to a well-studied close relative can provide a rudimentary validation of the calls as a ground truth. This process will confirm that at least most of a closely related fungal genome's coding sequences are predicted and shared by the gene calls for *Trichoderma*. Results for this processing can be produced with a simple BLAST search and apropriate cutoff values (i.e. query coverage, percent identity, E-score, etc.). While running BLAST is a simple process, the selection of a closely related organism is more difficult. One initial choice would be to work with *Saccharomyces cerevisiae*, or bakers yeast, as it is extremely well studied and would be considered a model organism, similar to *Arabidopsis* and *Mus musculus*. However, *Saccharomyces cerevisiae* diverged evolutionarily millions of years ago, which may make it a poor candidate for a comparative analysis. The second candidate considered for comparison is *Fusarium avenaceum*, as it is also well studied an more closely related to *Trichoderma* than yeast, and has a genome similar in size to that of *Trichoderma* species, at roughly 40Mb. Finally, comparison of assemblies and predicted genes to another *Trichoderma* strain is a reasonable approach. In this case, *Trichoerma atroviride* was selected as it is not included in the species used in the gene prediction portion of this analysis. From these assemblies, the RefSeq proteins (queries) from NCBI will be used in a tblastn search against each genome sequence from DC1, Tsth20, *T.reesei*, *T. harzianum*, and *T. virens* (subjects). Resulting BLAST hits will then be filtered based on suitable alignment coverage and identity, which will be determined by the reference sequence being considered. Total number of BLAST hits reported for each organism will provde information about completeness of assemblies and overall coverage of the gene/coding sequence space in the query sequences (need to be clear with language used for blast subjects and query). In addition, BLAST hits will be analysed with the region identification approach to identify coverage of protein and coding sequences in relation to gene predictions generated previously.

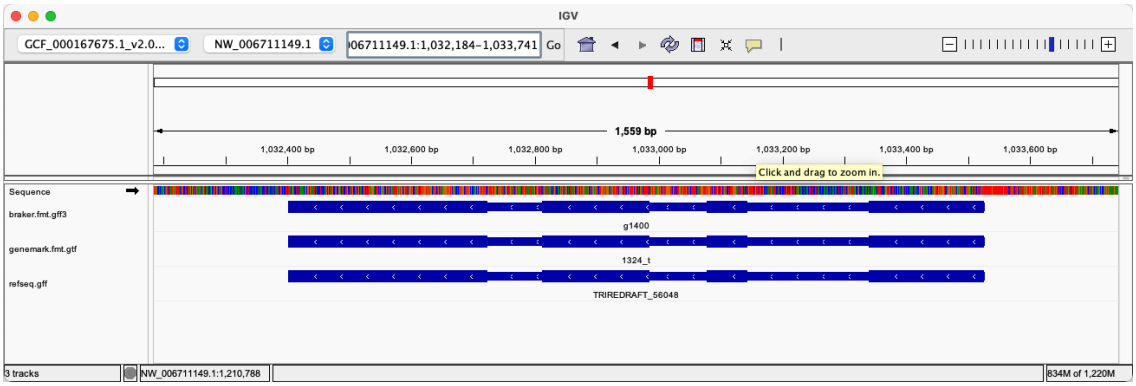
4.5.6 BUSCO Analysis

Another method for assessing the completeness of a set of predicted genes is Benchmarking Universal Single-Copy Orthologue (BUSCO) analysis[**?**]. BUSCO analysis is similar to analysis of overlapping or shared gene content with a close relative in that we are comparing the predicted gene sets to an existing standard or reference. With BUSCO analysis, the reference set has a far more strict definition. The datasets used for BUSCO analysis are based on single-copy orthologs generally found in an genome of interest. What this means is that BUSCO searches for single-copy genes that should be present in an organism based on the

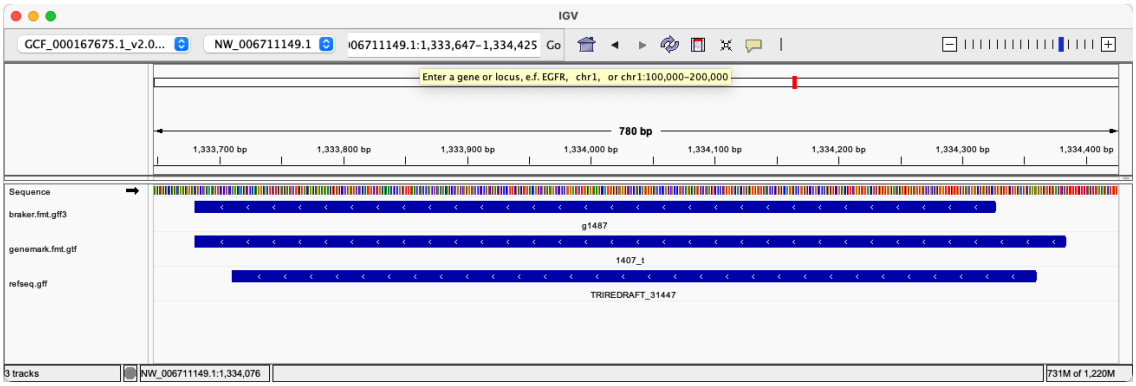
database selected for analysis. As an example in fungi, if one were interested in assessing the completeness of their annotated genes in a similarly related fungi, there should be a set, or subset, of single-copy genes present in the new annotation that are expected to be present after evolutionary divergence(...). This can be thought of as similar to a 'core' gene set. Results from BUSCO analysis are typically reported in a percentage of the gene set included in the BUSCO dataset. Percentages of single and duplicated hits are reported as well. While a high reported coverage of the BUSCO dataset is considered good, it is not fully indicative of excellent gene finding performance

4.5.7 Comparative Genomics

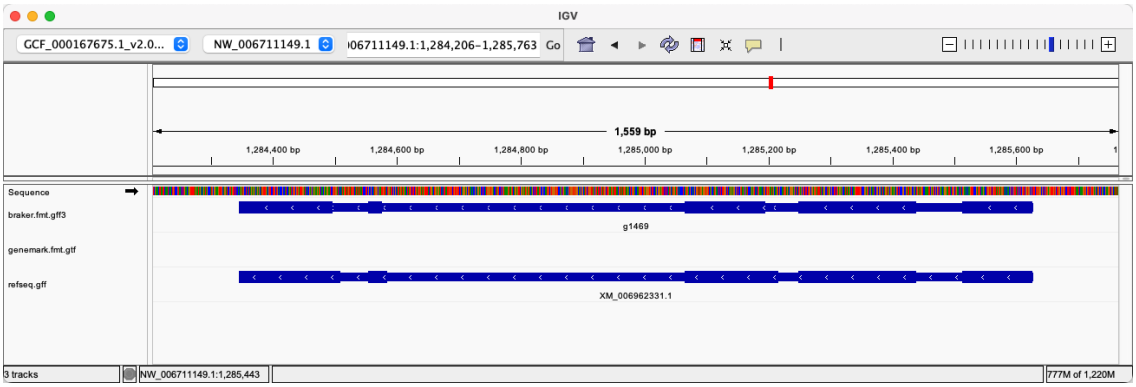
With the data produced by this research, it is possible to perform som commparative genomics (time permitted), mostly related to the assemblies generated during this work along with the RefSeq genomes included from NCBI. Mummer is a potential tool to use for all to all genome alignments, although there may be difficulty in the ordering of contigs/scaffolds/chromosomes when performing thes alignments. This work is not necessarily required but would be interesting from a biological perspective to identify rearrangements, inversions and such.



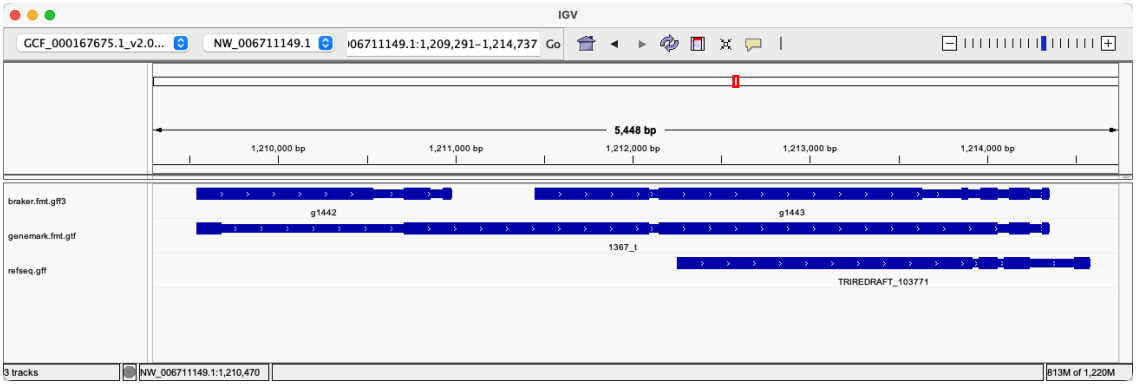
(a) Complete Agreement



(b) Start/Stop Disagreement



(c) Missing Prediction



(d) Complete Disagreement

Figure 4.2: Several visual examples of regions using IGV. **a)** Region where all prediction tools are in agreement. **b)** Prediction tools agree that gene is present, but not on the exact start and/or stop positions. **c)** A region where one tool does not predict a gene while the others do. **d)** A region with a combination of disagreeing predictions.

5 Results

5.1 Assemblies of DC1 and Tsth20

For general assembly metrics of DC1 and Tsth20, the QUAST tool was used. Results from QUAST are shown in figure 5.1, from which we can make several observations. In DC1 and Tsth20, the total contig counts are an order of magnitude smaller when compared to the other NCBI RefSeq assemblies, inidicating highly contiguous assemblies from nextDenovo and nextPolish. This is likely due to the use of long-read sequencing used in the assemblies of DC1 and Tsth20. The total assembly lengths are similar, ranging from 38Mb to 42Mb, except in the case of *T. reesei*, which is known to have a significantly smaller genome length [**?**] at roughly 33Mb. The largest contig size for each assembly varies greatly. DC1 and Tsth20 have the largest contigs of all assemblies being considered, which is again likely due to the inclusion of long-read sequencing data in the assembly process. The N50 values for all assemblies are above 1Mb, with DC1 and Tsth20 N50s being at minimum three times larger than others assemblies. Results from this table indicate that the assemblies of DC1 and Tsth20 are more contiguous than the assemblies of *Trichoderma reesei, harzianum and virens* also considered in this analysis. While contiguity is not the sole indicator of genome quality, it does provide confidence in the quality of the input data and resulting assemblies. In general, the assemblies of DC1 and Tsth20 are of similar length to existing *Trichoderma* assemblies and the number of contigs reported match the number of 'chromosome' scale contigs reported in other work.

During initial investigation of the sequences used as input to the assembly process, we observed that the reads contained abnormal ratios of GC content. To see if this observation extended to the assemblies as well, 250 bp sliding windows were used to calculate GC content for all assemblies included in this analysis. The

Strain	Total Contigs	Total Length	Largest Contig	GC%	N50	L50
DC1	8	38.6 Mb	11.49 Mb	47.97	5.69 Mb	3
Tsth20	7	41.58 Mb	8.02 Mb	47.33	6.52 Mb	3
T. harzianum	532	40.98 Mb	4.08 Mb	47.61	2.41 Mb	7
T. virens	93	39.02 Mb	3.45 Mb	49.25	1.83 Mb	8
T. reesei	77	33.39 Mb	3.75 Mb	52.82	1.21 Mb	9

Table 5.1: General assembly metrics produced by QUAST (a genome quality assement tool).

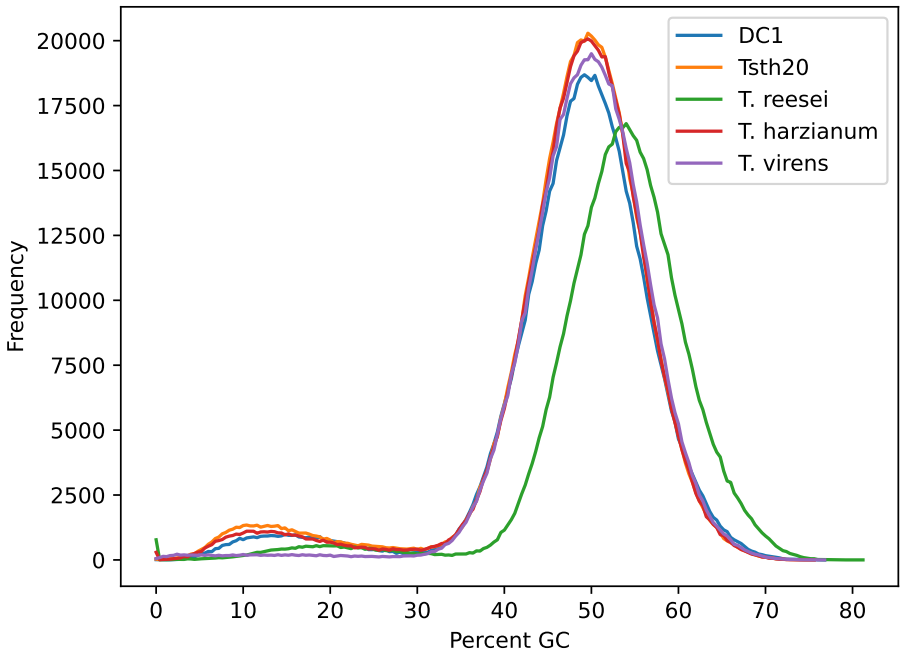


Figure 5.1: Plots showing the frequency of GC values calculated from sliding windows for each assembly.

results of this analysis are shown in figure 5.1. Of the included assemblies, anomalous GC content in the form of AT-rich sequences were identified in DC1, Tsth20, *T. reesei* and *T. harzianum*, with *T. virens* deviating from the other assemblies showing very few AT-rich windows. Anomolous GC content is visualized on the left tails of the distributions with a local peak around sequences containing 10 percent GC content. In addition to the confirmation of increased AT-rich sequence content in most assemblies, it appears that the distribution of GC content in *T. reesei* differs from the other assemblies. The curve of GC content for *T. reesei*, visualized in green in 5.1, lies farther to the right, indicating more GC content, or fewer AT-rich windows, in its assembly. While the left tail of the curve also shows an increase in AT rich sequence composition, it is shifted farther right than other *Trichoderma* assemblies. Investigation of these anomalous regions is continued in section ...

5.2 Profiling Gene Finding Tools

While Braker2, GeneMark and RefSeq all provide lists of possible genes for a provided reference, the implementation of each tool is different, requiring more or less effort to install and run depending on the tool used. The computing platform used in this research is hosted and managed by University of Saskatchewan services, which is modelled around the HPC platform used by the Digital Research Alliance of Canada and software is managed similarily.

First, we will briefly discuss the RefSeq annotation process. RefSeq annotation is only applied to data that is submitted to NCBI. The RefSeq Eukaryotic Genome Annotation Pipeline[9] is a genome annotation process developed and maintained by NCBI. The pipeline is not directly publicly available to public users, and requires submission of data to NCBI. Once data is submitted to NCBI, the RefSeq annotation pipeline may be applied upon request only if the genome is the highest quality assembly for the species in question or if the genome is of significant interest to the scientific community, limiting reach of the annotation process to many users. The pipeline supplies existing RNAseq, CDS and protein sequences to NCBI's in-house gene prediction tool Gnomon, which produces trained models for gene prediction. While the tools used for alignment and processing of supporting sequence information are listed, the inner workings of Gnomon are not well documented, at least from the public perspective, and I was unable to find Gnomon in any compilable or executable form during my search. Recreation of the RefSeq pipeline would prove extremely challenging if not impossible without supporting information. Run times for this pipeline are difficult to determine due to the hidden nature of the pipeline, unknown compute resources and varying quantities of data used. The RefSeq annotation process produces comprehensive outputs, including CDS sequences, translated CDS, RNA from genomic sequences, proteins, feature counts and tables, and finally GFF and GTF formatted annotation files for these features.

Next we will discuss the handling, installation, and running of both Braker2 and GeneMark packages. Several points will be discussed for each tool, with those points being sourcing and downloading, components and prerequisites, installation, execution, and output. GeneMark[6] is a gene finding tool developed by the Georgia Institute of Technology with packages prepared for Linux and MacOS. It is provided as licensed product in the form of a package which can be downloaded from their website after submitting a form. Once the necessary information is submitted, the user is provided with a key that must be placed in the appropriate location once the software is downloaded and unpacked. The core controlling methods of GeneMark are written in Perl, accompanied by several Python scripts and compiled exectuables. GeneMark was tested by the developers with Perl version 5.10, and Python 3.3. A number of Perl dependencies are also required, which can be installed via YAML. The user will have to know which implementation of GeneMark they are wanting to use for their application, as GeneMark has several variations it can run depending on the desired application. In this work, the GeneMark-ES variant of GeneMark was executed, as it is the self-training *ab initio* GeneMark method for eukaryotic organisms .Options required by GeneMark at runtime are documented in the help message, and simple enough that any user with familiarity of bioinformatics tools should be able to run GeneMark, although documentation for use is only provided by the help message when running the program and not online. In regards to run-time, running the GeneMark-ES pipeline on DC1 with 56 threads finished in 16 minutes. Upon completion, GeneMark produces a GTF or GFF file of predicted genes as well as a number of other outputs related to the run.

Braker2[2] is hosted on GitHub as a repository that receives relatively frequent updates with Braker3 being released while working on this thesis. Braker is maintained by Katharina Hoff from the Univer-

sity of Greifswald and is available under the Open Source Artistic License. Installation of the repository is a straightforward pull from GitHub. As with GeneMark, Braker2 uses a combination of Perl, Python and other exectuables in its regular use. Downloading the repository itself is not enough for execution, as Braker2 relies on a number of dependencies and bioinformatics tools including Perl and (Perl dependencies), Augustus, BamTools, BedTools, GeneMark, StringTie, GFFRead and a few others. Manual installation of these dependencies would be difficult, time consuming and in general advised against. In this case, many of Braker2's requirements are satisfied by modules already included in the environment, making installation relatively simple if you know the ins and outs of the Digital Research Alliance of Canada's software stack. This case still required installation of some Perl modules in addition to loading necessary modules. Alternatively, one could use a package manager like Anaconda or Minoconda to handle installation of packages. This is perfectly reasonable, but also requires knowledge specific to Anaconda, which can be complicated and frustrating for users with little software management experience. The newly released Braker3, also includes a containerized version of the software, allowing users to build and execute Braker3 with ease in its own environment. Once installation is finished, the Barker2 pipeline is relatively straightforwad to run as well, with excellent documentation included both online and through the built in help message. In this research, the Braker2 pipeline was run in two modes. The first mode is a training mode, where sequence files are supplied to Braker2 to train a gene-calling model. The training and gene finding steps are run as part of the same command, and a model built using the training data is saved for bookkeeping and future use. The training was performed using roughly 145 million Illumina paired end RNAseq reads on the *Trichoderma reesei* genome. The training, including RNAseq alignments, and gene finding pipeline in this case took 1 hour and 17 minutes using 60 threads. Applying the Braker2 trained gene finding model to DC1 with 60 threads took 21 minutes to complete. Run times will of course vary depending on processing power available to the end user, but in the case of *Trichoderma* genomes, users can expect quick results with relatively little computing power. Once annotation is complete, Braker2 produces a GFF file containing predicted genes along with CDS sequences and amino acid sequences their protein products. Braker2, when trained with RNAseq data, includes an option to predict UTR regions, but it is experimental and is not performed by default and is not included in these runs.

In summation, Braker2 and GeneMark are not direct plug-and-play software packages. Users should expect to encounter issues when getting these programs running in addition to normal downloading and unpacking of software packages so some expertise is recommended. Neither Braker2 nor GeneMark require users to compile software, however Braker2's dependencies may require additional compilation and attention. Once installed, both tools are relatively simple to use with documention available for both on the commandline and excellent documentation available for Braker2 on their GitHub page. Outputs from both tools are similar although Braker2 has the ability to output coding and amino acid sequences for downstream processing. Both tools run in reasonable amounts of time, where in the case of smaller genomes such as *Trichoderma*, users can expect results within a few hours to a day depending on number of computing cycles available to

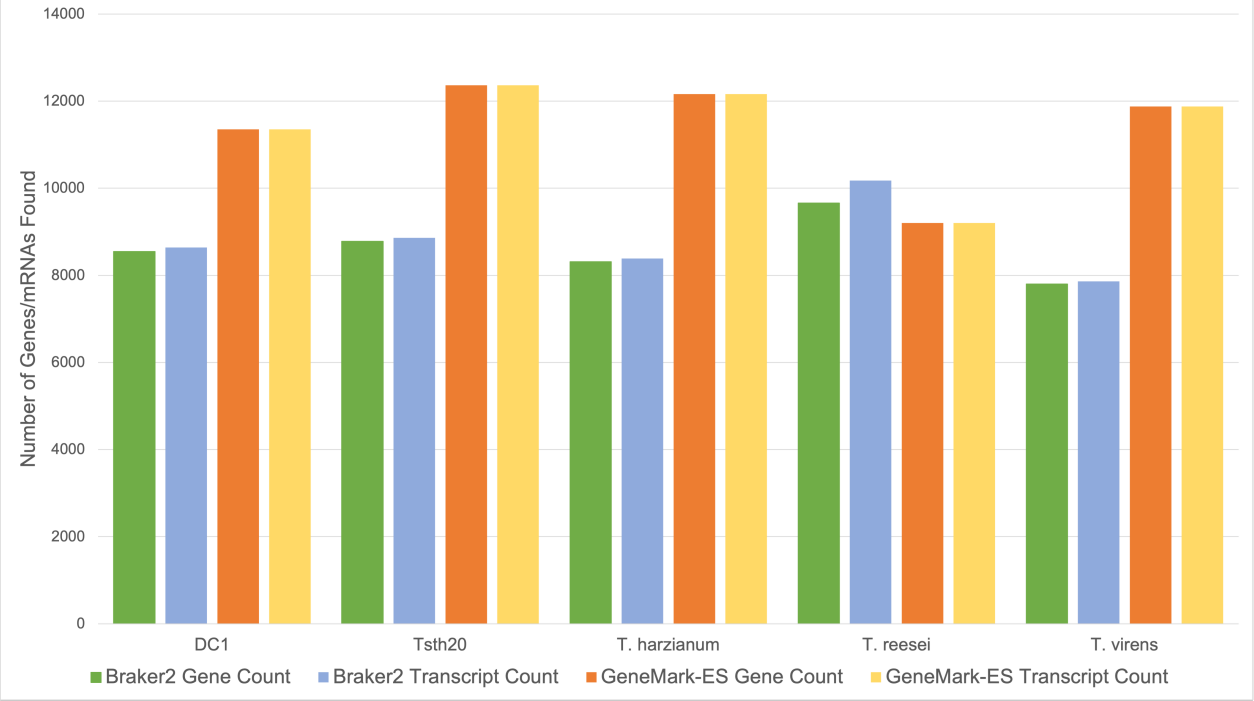


Figure 5.2: Shows the counts of genes and mRNAs found by each gene finding tool for each genome assembly considered.

them.

5.3 Initial Gene Finding Results

In figure 5.2, we see a summary of total genes and mRNAs predicted by each of the selected tools. An immediate trend can be seen in this data. The total predicted features for Braker2 are significantly lower than those from GeneMark in all assemblies except for *T. reesei*. This may be due to Braker2 using RNAseq data from *T. reesei* during its training process, which we will cover in the discussion section. Another observation can be seen when comparing the predicted genes and predicted RNAs for the same tool when applied to the assemblies. GeneMark does not appear to identify any additional isoforms, only reporting the entire gene structure. Braker2 does identify isoforms, although very few of them. This may be related to the training set provided to the training process, although this is not yet confirmed. Overall, it appears that GeneMark regularly predicts a higher number of features when applied to these assemblies.

One important ascpect to consider when looking at the output of different gene prediction tools is the distribution of sequence lengths predicted by any given tool. Lengths of possible sequences can vary widely, ranging from small non-coding RNAs, which can be less than 200 nucleotides in length, up to the largest genes which cover more than two kilobases. Due to this wide variation in possible sequence lengths, it is possible that different prediction tools could produce different distributions of predicted sequence lengths. This is important if researchers are interested in small non-coding RNAs or atypically large genes. This

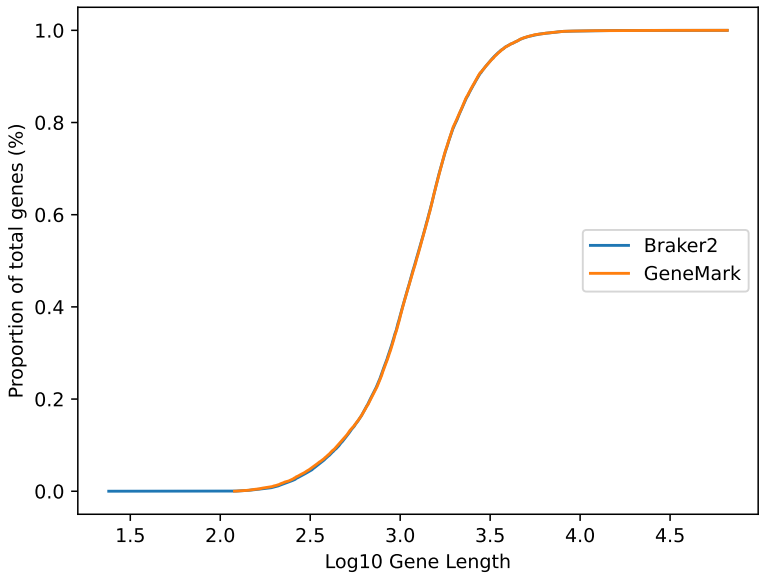
section will investigate the coding sequence lengths predicted by Braker and GeneMark for DC1 and Tsth20 while the RefSeq assemblies will also include the RefSeq annotation.

5.4 Distribution of Predicted Gene Lengths

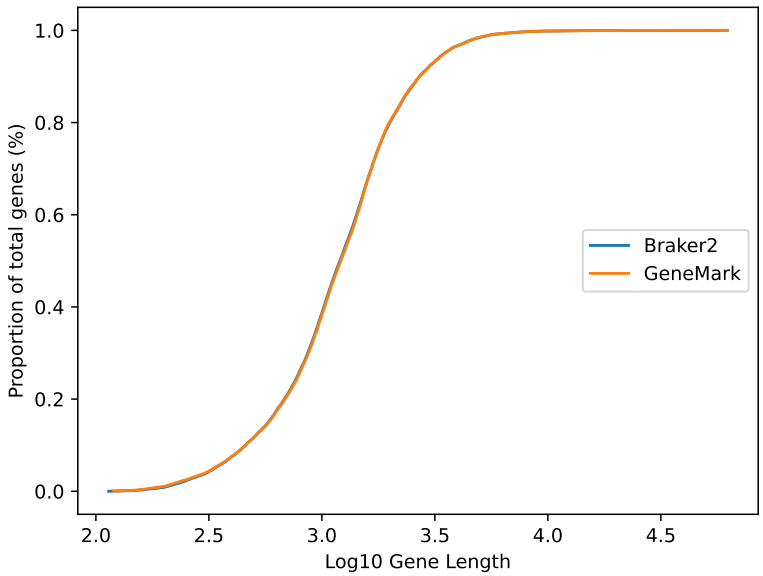
To better understand and compare the distributions of gene lengths predicted by Braker2, GeneMark, and RefSeq, the cumulative density function for the lengths of CDS sequences for each gene finding tool are shown in figure 5.3. The $\log_{10}$ values of gene lengths were used for a better visulazation of the distributions. In DC1, the curves from Braker2 and GeneMark follow each other closely, with the only variation being genes of short length, where Braker2's curve extends beyond that of GeneMark, indicating that Braker2 predicts more shorter genes than GeneMark. In the case of Tsth20, the curves are nearly identical. In *T. reesei*, we see disagreement in the curves for shorter genes, with Braker2 appearing to predict a greater amount of shorter genes than GeneMark and RefSeq. The upper portions of the curves trend toward similar predicted gene lengths. In *T. harzianum*, RefSeq deviates from GeneMark and Braker2, predicting more genes of short length, while Braker and GeneMark appear to be in near-complete agreement. Finally, in *T. virens*, we see the RefSeq curve predicting shorter genes once again, although the deviation is not as drastic as in *T. harzianum*. From these plots, we can say that visually, gene finding tools appear to predict different lengths of genes. We also observe that Braker2 predicts the shortest genes of all three gene finding methods.

To confirm these observations statistically, two-sided two-sample Kolmogorov-Smirnov tests were performed using the $\log_{10}$ transformed gene lengths and are presented in table 5.2. In the cases of DC1 and Tsth20, we see that Braker2 and GeneMark do not produce statistically different lengths of genes, which is interesting considering that the Braker2 includes experimental evidence for another assembly while GeneMark does not. In *T. reesei*, while RefSeq and GeneMark do not reject the null hypothesis when compared, Braker2's predicted gene lengths are significantly different than both RefSeq and GeneMark, which is also confirmed visually in the CDF plots above. The same cannot be said for *T. harzianum* and *T. virens*, where RefSeq is significantly different from both GeneMark and Braker2, which are not significantly different from eachother. It is also notable that RefSeq and GeneMark predict similar gene lengths in *T. reesei* but not in *T. harzianum* and *T. virens.*

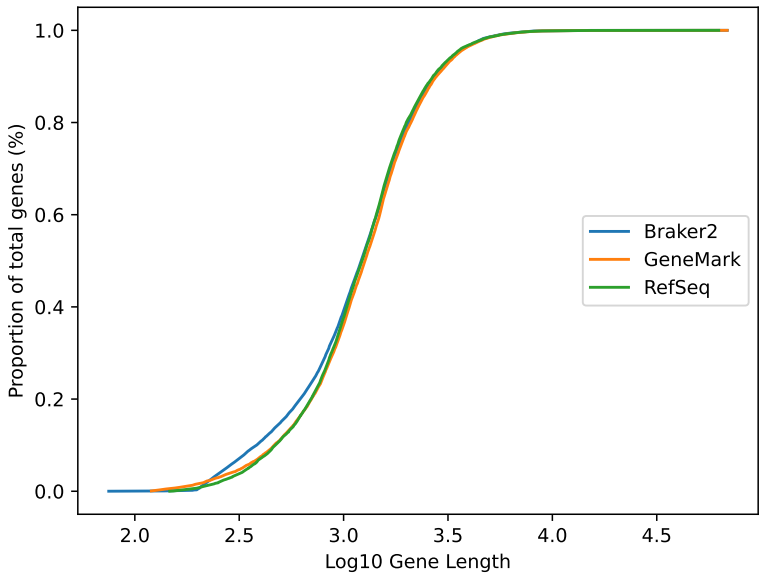
It can clearly be stated that these gene finding tools predict different lengths of genes, paritcularly in *T. reesei, harzianum T. virens.* Why that may be the case is difficult to answer without deeper invesitgation. There is clearly an underlying difference between the gene finding tools, in particular when training data is introduced. We observe that when gene finders are applied to the assemblies from which their training data originated, they tend to predict shorter genes. Conversly, we observe that when a gene finding tool trained with experimental evidence is applied to a different *Trichoderma* assembly, the lengths of predicted genes do not significantly differ from an *ab initio* gene finder like GeneMark, except in the case of RefSeq's predictions on *T. reesei.*



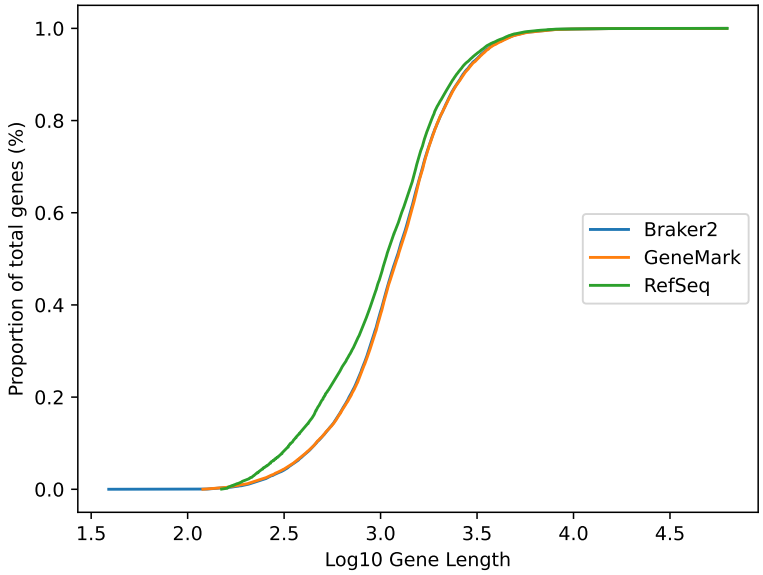
(a) DC1



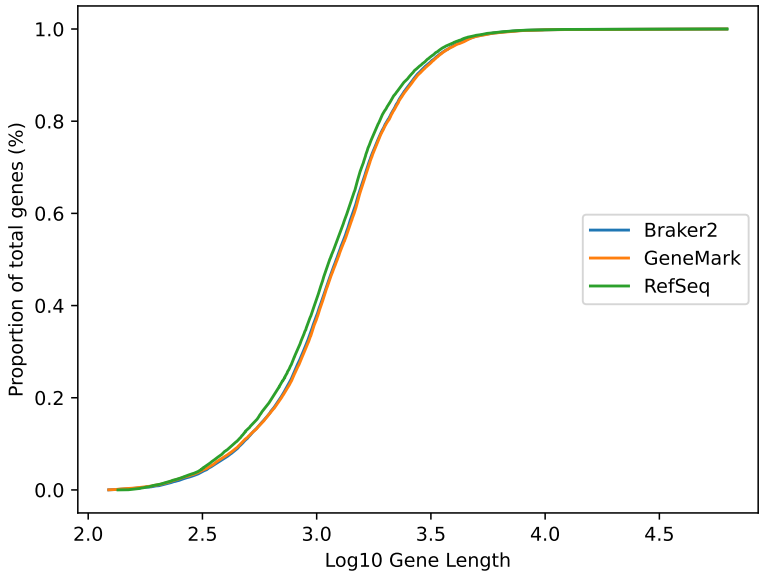
(b) Tsth20



(c) *T. reesei*



(d) *T. harzianum*



(e) *T. virens*

Figure 5.3: Plots of the cumulative density function for CDS lengths predicted by each gene finding tool.

Genome	Tool #1	Tool #2	*P*-value
DC1	Braker2	GeneMark	0.999
Tsth20	Braker2	GeneMark	0.965
T. reesei	Braker2	GeneMark	9.481^{-07}
T. reesei	GeneMark	RefSeq	0.002
T. reesei	Braker2	RefSeq	1.340^{-07}
T. harzianum	Braker2	GeneMark	0.863
T. harzianum	GeneMark	RefSeq	4.313^{-52}
T. harzianum	Braker2	RefSeq	4.674^{-55}
T. virens	Braker2	GeneMark	0.635
T. virens	GeneMark	RefSeq	7.352^{-12}
T. virens	Braker2	RefSeq	1.794^{-09}

Table 5.2: Table of *P*-values from two-sided two-sample Kolmogorov-Smirnov tests between gene finding tools.

Reference	Ref. Proteins	DC1	Tsth20	*T. reesei*	*T. harzianum*	*T. virens*
T. atroviride	11807	11552	11080	10601	11081	11078
Fusarium	13312	10327	10429	10064	10434	10490
S. cerevisiae	6014	3537	3517	3445	3509	3500

Table 5.3: tBLASTn hits from reference protein sequences to selected assemblies of intereset. Hits are reported if the alignment length is greater than 30% of the reference protein length and if 30% of the aligned length have identical matches.

5.5 BLAST Results

Results from the T-BLAST-N runs are presented in table 5.3. Initial BLAST results appear promising for both the *T. atroviride* and *Fusarium* datasets. All assemblies considered contain at minimum 89% of the reference protein sequences in the case of *T. atroviride* and a minimum of 75% in the case of *Fusarium*. Following the trend of gene calls, *T. virens* returns the fewest hits of the selected assemblies in all cases while the other assemblies report a similar number of hits as each other. In the case of *S. cerevisiae*, a minimum of 57% of reference proteins matched. These results provide rough validation that the assemblies contain potential for protein coding sequences. Successful hits from this process will be retained and used in the region identification process as validation for gene calls from selected tools.

5.6 BUSCO Results

The results of BUSCO analysis using the fungal subset provided by BUSCO are presented in table5.4. Results from BUSCO indicate that all gene sets considered in this analysis have a BUSCO completeness of 99.2% or higher, with a maximum Completeness of 99.9% for some gene sets. In general, Braker2 and RefSeq have the most BUSCO complete sets of gene predictions fo the three tools considered. Interestingly, Braker2 produces far more duplicated BUSCO matches than both GeneMark and RefSeq. All tools considered have very low fragmented and missing values.

5.7 Region Identification

The region identification process begins with identification of regions sharing gene calls from each tool. For this project, we have classified regions as complete, partial or singleton. A complete region is a set of overlaps which contains a feature from each tool considered in the region finding process. A partial region is a set of overlaps which includes more than one but not all tools considered. A singleton is a region in which a feature from only one tool is present. Table 5.5 displays the results from the region finding process when applied to only the features of type 'gene' predicted by each tool.

Strain	Complete	Single	Duplicated	Fragmented	Missing	No. markers
DC1	99.5	80.2	19.3	0.1	0.4	758
Tsth20	99.9	81.7	18.2	0.0	0.1	758
T. harzianum	99.7	80.2	19.5	0.0	0.3	758
T. virens	99.8	79.0	20.8	0.1	0.1	758
T. reesei	99.9	85.5	14.4	0.1	0.0	758

(a) Braker2

Strain	Complete	Single	Duplicated	Fragmented	Missing	No. markers
DC1	99.2	98.8	0.4	0.3	0.5	758
Tsth20	99.8	99.1	0.7	0.0	0.2	758
T. harzianum	99.6	98.9	0.7	0.0	0.4	758
T. virens	99.7	99.2	0.5	0.1	0.2	758
T. reesei	99.6	99.5	0.1	0.0	0.4	758

(b) GeneMark

Strain	Complete	Single	Duplicated	Fragmented	Missing	No. markers
T. harzianum	99.9	99.2	0.7	0.0	0.1	758
T. virens	99.5	98.8	0.7	0.3	0.2	758
T. reesei	99.8	99.5	0.3	0.0	0.2	758

(c) RefSeq

Table 5.4: Results from BUSCO using the fungal analysis option organized by gene finding tool.

Assembly	Regions (total)	Complete Agreement	Partial Agreement	Singletons
DC1	11269	8483	N/A	2786
Tsth20	12272	8737	N/A	3535
T. reesei	9823	8282	557	984
T. harzianum	13388	8009	3314	2065
T. virens	12045	7537	3715	793

Table 5.5: Counts of regions identified in total and total number of regions where a prediction from each individual tool was found. Partial agreement values for DC1 and Tsth20 are set as N/A as there were only two tools in consideration.

Assembly	Braker Partial	Braker Single	GeneMark Partial	GeneMark Single	RefSeq Partial	RefSeq Single
DC1	N/A	3	N/A	2783	N/A	N/A
Tsth20	N/A	6	N/A	3529	N/A	N/A
T. reesei	387	533	423	175	304	276
T. harzianum	38	2	3309	209	3281	1854
T. virens	30	0	3700	162	3700	631

Table 5.6

The results of the region finding process when applied to gene calls show a mix of agreement and disagreement between the tools considered here. While regions of complete agreement make up the majority of regions in all assemblies, there are more partial agreements and singletons than one would expect under the assumption that gene finding tools are equal. Both DC1 and Tsth20 have a large number of singleton regions present in comparison to the RefSeq datasets.

The composition of gene calls from partial and singleton regions will help provide an understanding of each tools performance when looking for unqiue gene predictions. Table 5.6 displays counts of genes for each gene finding tool that are 'unique', or are not shared in a complete region.

From table 5.6, we can see that Braker's ability to predict unique or semi-unique genes is weaker than that of GeneMark and RefSeq in all assemblies except in the case *T. reesei*. GeneMark predicts significantly more unique genes than Braker for the assemblies of DC1 and Tsth20 and a similar number of semi-unique and unique genes as RefSeq except in the case of *T. harzianum* where RefSeq predicts far more unique genes.

5.8 Genes in Regions of Anomalous GC Content

Evaluating gene finder performance in regions of anomalous GC content is one of the key topics of this research. One simple way to evaluate performance is whether or not gene finding tools predict genes uniformly throughout a given sequence. Biologically, we know that regions of anomalous nucleotide composition are less likely to contain coding sequences than typical genomic regions, leading us to the problem of first identifying predicted genes in standard and anomalous regions. After identifying low GC segments within each assembly, we can include them in the region identification method. Counts from region identification with the inclusion of anomolous GC content are presented in table 5.7.

To better understand the performance of each tool in identifying genes in repetitive sequences, we will investigate the composition of the partial and singleton regions. Table **??** shows the the number of genes identified by each tool in those regions.

From this result, we classified predicted genes into two classes; genes in regions with normal GC content, and genes in regions with anomalous content. In this case, anomalous content is defined as a window of

Assembly	Total Regions	Complete Agreement	Partial Agreement	Singletons
DC1	11856	10	N/A	20
Tsth20	12507	2	N/A	9
T. reesei	10428	25	19	115
T. harzianum	14622	26	45	548
T. virens	12121	8	11	17

Table 5.7: Total regions identified each assembly followed by counts of regions (both agreement and singletons) in regions of anomalous GC content.

Tool	DC1	Tsth20	*T. reesei*	*T. harzianum*	*T. virens*
Braker2	9.56^{-181}	1.14^{-259}	2.68^{-96}	4.05^{-140}	1.35^{-35}
GeneMark	5.12^{-216}	0.0	5.66^{-49}	5.37^{-219}	5.31^{-35}
RefSeq	N/A	N/A	1.29^{-49}	2.44^{-205}	7.40^{-33}

Table 5.8: p values produced from a two-sided binomial test for each combination of tool and assembly.

genomic sequence containing a percent GC composition of 28% or lower. This number was chosen based on the plots of GC content presented in the assembly section of the results. After classifying predicted genes, two-sided binomial tests were performed with the null hypothesis being that predicted genes are distributed uniformly throughout an assembly. Framed differently, we expect the sum of genes predicted in both regular and irregular regions to be proportional to the sum of lengths of those regions, respectively. This is not the case as demonstrated in table 5.8.

5.9 InterProScan as Supporting Evidence for Predicted Genes

Pfam hits from InterProScan analysis are presented in table 5.9, from which we can identify one major trend. In general, between 65 and 75 percent of proteins predicted by both Braker2 and GeneMark contain a match to a Pfam entry. This is promising performance for the gene finders as the RefSeq annotations demonstrate similar proportions of Pfam hits to predicted proteins. The total counts may be decieving however, as predictions from Braker2 may result in more than one protein product per gene, whereas in the case of GeneMark, only one protein is produced per gene model. From visual inspection of Pfam hits mapped back to the references, it would appear that in general, when gene finders agree that a gene is present, InterProScan reports the same Pfam match in all three predictions. An example of agreement between Braker2, GeneMark, RefSeq and InterProScan is shown in figure 5.4. It is important to note that the positioning of the Pfam match does not indicate the exact position in the gene sequence, but provides an

indication of the presence of Pfam matches. Pfam matches are offset from the start of the gene based on the start and end position of the Pfam match in the protein sequence. For example, if a Pfam match has a start position 10 amino acids into the protein sequence, the corresponding start position in the resulting GFF is 10bp downstream from the start of the gene.

Assembly	Braker Proteins	Braker Pfam Hits	GeneMark Proteins	GeneMark Pfam Hits	RefSeq Proteins	RefSeq Pfam Hits
DC1	14479	10676	11354	8416	N/A	N/A
Tsth20	15546	11389	12373	9168	N/A	N/A
T. reesei	11704	8471	9196	6990	9111	6964
T. harzianum	15408	11370	12164	9061	14065	9293
T. virens	15062	11249	11866	8871	12383	9062

Table 5.9: Table with counts of predicted genes with Pfam annotations from InterProScan

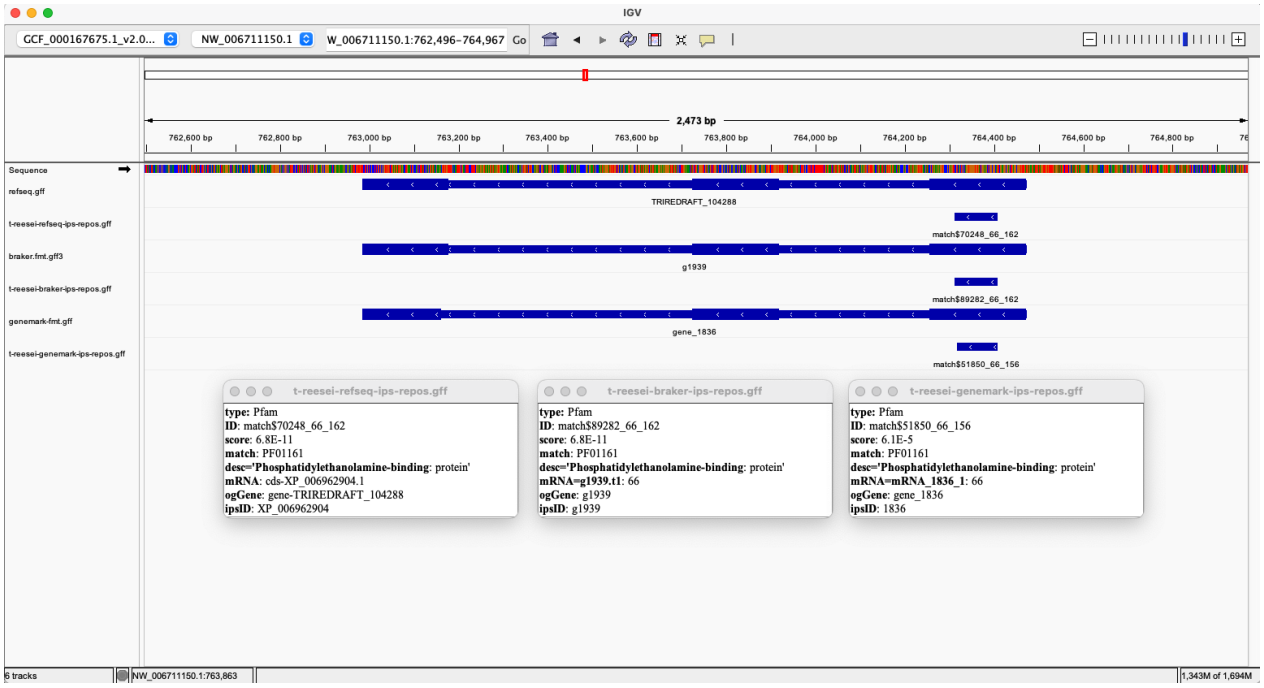


Figure 5.4: An IGV capture showing complete agreement between gene finders for both gene model and protein Pfam hits

While cases of complete agreement are abundant, cases of disagreemnt also exist and in strange forms. In many cases, while the gene finders agree on the presence of a gene, only the RefSeq protein product contains a match to the Pfam database. Why this may be the case is unclear, and may warrant further investigation. There are also many cases in which InterProScan reports the same Pfam hits for individual proteins, but the gene models in the region do not agree. There are even cases such as the region shown in figure 5.5, where Braker2 and GeneMark agree that two genes and their associated proteins and Pfam matches are separate,

but RefSeq only reports one gene with multiple Pfam hits. There are also several cases where two tools are in agreement with proteins containing Pfam hits while another is not. Even more interesting are cases such as the one shown in figure 5.6, in which Braker and GeneMark predictions contain Pfam matches while RefSeq does not report a gene at all. This may be the due to experimental data used in the RefSeq training process or a result of curation. Regardless of the tool, these cases demonstrate well that gene finders are not always in agreement, to the point that predictions are not present even though protein products from other gene finders contain known Pfam matches.

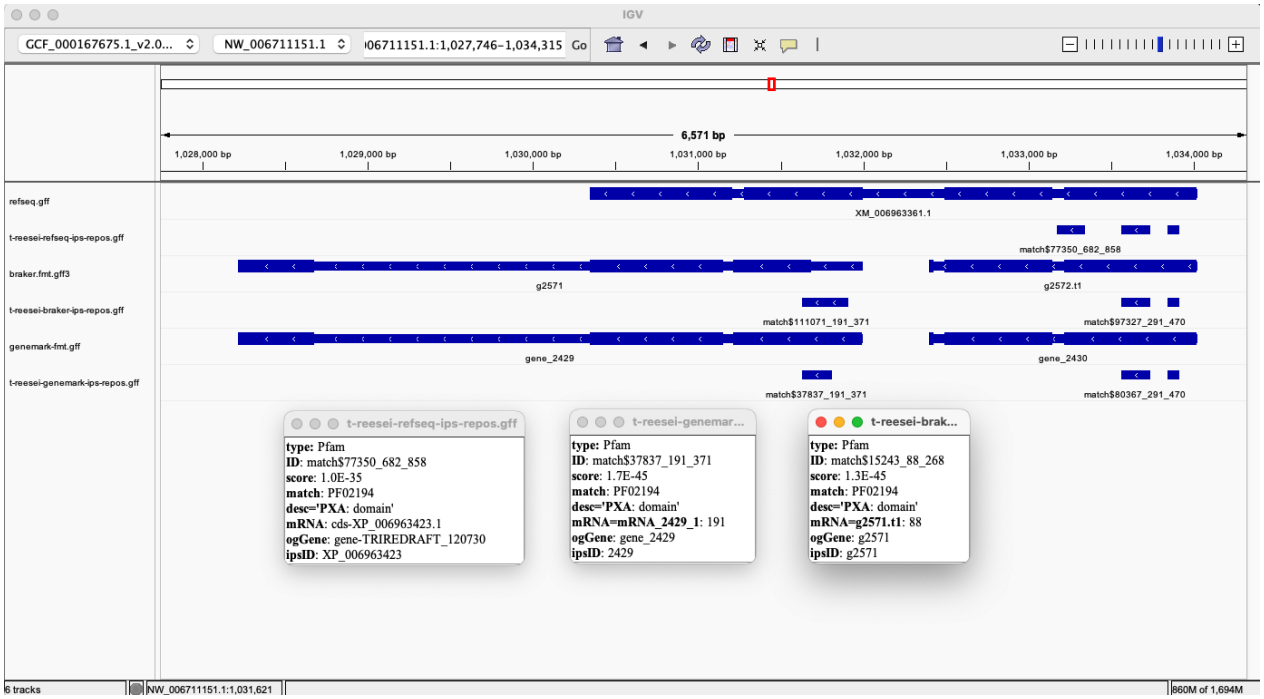


Figure 5.5: An IGV capture showing Braker2 and GeneMark reporting two genes and their resulting proteins and Pfam hits as separate, while RefSeq reports one gene, one protein and three Pfam matches.

In summary, the protein products from Braker2 and GeneMark predictions do contain matches to the Pfam database. The proportions of matches to total proteins are similar to that of the RefSeq annotation, sitting between 65 and 75 percent. While Pfam hits to Braker2, GeneMark and RefSeq proteins generally agree, we do observe regions in which there is disagreement in serveral forms.

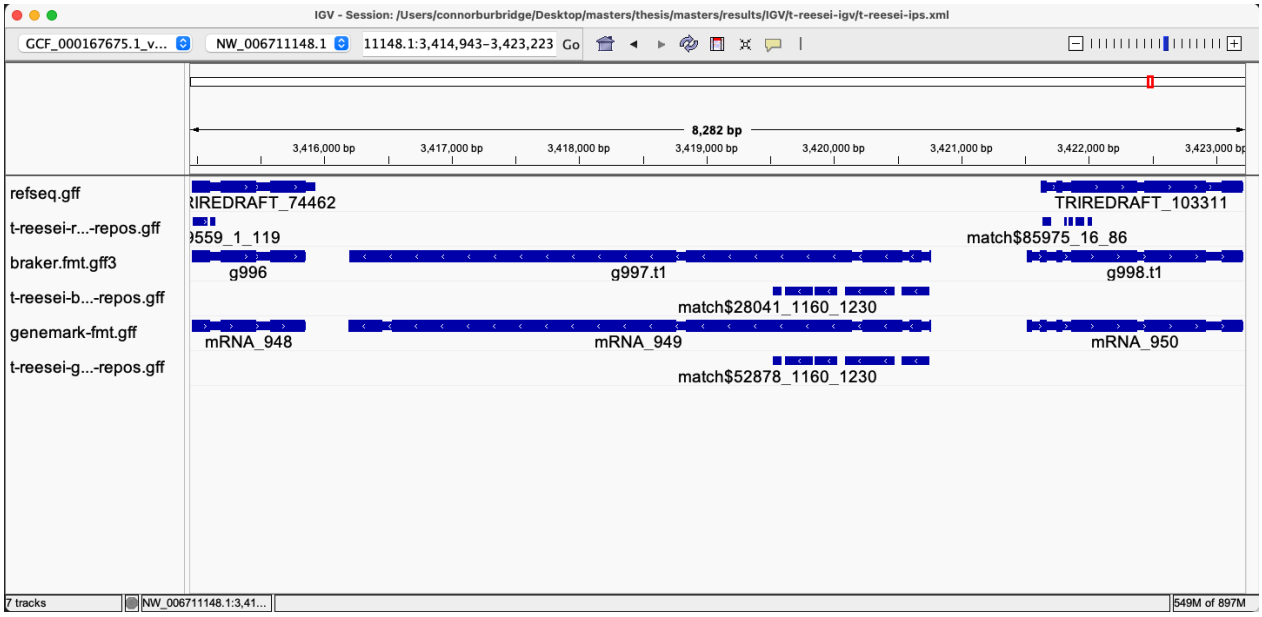


Figure 5.6: An IGV capture showing a scenario where GeneMark and Braker2 agree on a gene model with supporting Pfam evidence and RefSeq does not report any gene.

5.10 Conclusions

References

[1] Jill Adams. Complex Genomes: Shotgun Sequencing. *Nature Education*, 1(1):186, 2008.

[2] Tomáš Brůna, Katharina J Hoff, Alexandre Lomsadze, Mario Stanke, and Mark Borodovsky. BRAKER2: automatic eukaryotic genome annotation with GeneMark-EP+ and AUGUSTUS supported by a protein database. *NAR genomics and bioinformatics*, 3(1):lqaa108, mar 2021.

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

[4] Girum Fitihamlak Ejigu and Jaehee Jung. Review on the Computational Genome Annotation of Sequences Obtained by Next-Generation Sequencing. *Biology*, 9(9), sep 2020.

[5] Sara Goodwin, John D McPherson, and W Richard McCombie. Coming of age: ten years of next-generation sequencing technologies. *Nature Reviews Genetics*, 17(6):333–351, 2016.

[6] Alexandre Lomsadze, Vardges Ter-Hovhannisyan, Yury O Chernoff, and Mark Borodovsky. Gene identification in novel eukaryotic genomes by self-training algorithm. *Nucleic Acids Research*, 33(20):6494–6506, 2005.

[7] Vivien Marx. Method of the year: long-read sequencing. *Nature Methods*, 20(1):6–11, 2023.

[8] Niranjan Nagarajan and Mihai Pop. Sequence assembly demystified. *Nature Reviews Genetics*, 14(3):157–167, 2013.

[9] NCBI. NCBI Eukaryotic Annotation Pipeline, 2024.

[10] Miroslav Plohl, Nevenka Meštrović, and Brankica Mravinac. Centromere identity from the DNA point of view. *Chromosoma*, 123(4):313–325, aug 2014.

[11] Biancamaria Senizza, Fabrizio Araniti, Simon Lewin, Sonja Wende, Steffen Kolb, and Luigi Lucini. Trichoderma spp.-mediated mitigation of heat, drought, and their combination on the Arabidopsis thaliana holobiont: a metabolomics and metabarcoding approach. *Frontiers in Plant Science*, 14, 2023.

[12] R Keith Slotkin. The case for not masking away repetitive DNA. *Mobile DNA*, 9(1):15, 2018.

[13] Ioannis S Arvanitoyannis Theodoros H. Varzakas and Haralambos Baltas. The Politics and Science Behind GMO Acceptance. *Critical Reviews in Food Science and Nutrition*, 47(4):335–361, 2007.

[14] Todd J Treangen and Steven L Salzberg. Repetitive DNA and next-generation sequencing: computational challenges and solutions. *Nature reviews. Genetics*, 13(1):36–46, nov 2011.

[15] David J Winter, Austen R D Ganley, Carolyn A Young, Ivan Liachko, Christopher L Schardl, Pierre-Yves Dupont, Daniel Berry, Arvina Ram, Barry Scott, and Murray P Cox. Repeat elements organise 3D genome structure and mediate transcription in the filamentous fungus Epichloë festucae. *PLOS Genetics*, 14(10):1–29, 2018.

[16] Sheridan L Woo, Rosa Hermosa, Matteo Lorito, and Enrique Monte. Trichoderma: a multipurpose, plant-beneficial microorganism for eco-sustainable agriculture. *Nature Reviews Microbiology*, 21(5):312–326, 2023.