Research Questions

1.1 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).

1.2 *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?

1.3 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*?**

1.4 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.

1.5 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*?**

1.6 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?**

1.7 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?**

1.8 Similarity Searches of Predicted Genes with tblastn

1.9 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?**

1.10 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?**

1.11 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?**

Data and Methodology

2.1 Methodology Overview

The general processing steps performed in this work can be grouped into four processing stages, or workflows. A simplified overview of these stages and the flow of data between them is shown in Figure 2.1. In ascending order, each processing stage is discussed in more detail in Sections 2.2, 2.3, 2.4, and 2.5.1.

2.2 Sequence Processing and Assembly Workflow

The sequence processing and assembly workflow is shown in Figure 2.2. Processing steps are represented by boxes, results of interest generated by this work are represented by folders, and datasets that were not produced by this work are represented by cylinders. Other workflows that use datasets produced by this workflow are depicted as arrows outside of the brown sub-graph.

Illumina©™ [4] sequences were first trimmed and filtered using Trimmomatic [5] to remove adapter content and low-quality sequences, as low-quality sequences may affect the assembly process, resulting in fragmented and erroneous assemblies. The processed Illumina©™sequences and Nanopore©™ [22] sequences were then supplied to the hybrid genome assembly tool NextDenovo [15]. As Ilumina sequences are of extremely high accuracy, especially after trimming, the processed Illumina©™data was then re-used to polish the assemblies with NextPolish [14]. Polishing is used because long-read sequences used in the assembly process are less accurate, at approximately 90-95%, than Illumina sequences at 99.9%, so the Illumina data can be used to correct any assembled sequences that may contain base-call errors. The resulting assemblies were then analyzed to identify AT-rich genomic sequence, analyzed with QUAST for general assembly metrics, and then used as input in the gene prediction workflow.

The processing steps Trimmomatic filtering, NextDenovo assembly, NextPolish Polishing and QUAST assembly assessment are described in Section 2.2.1. The AT-rich sequence identification process is described later in Section 2.5.2.

2.2.1 Sequence Pre-processing and Assembly

Genomic sequences used for assembly were generated for DC1 and Tsth20 using Nanopore©™ [22] and Illumina©™ [4] sequencing technologies by Brendan Ashby at the Global Institute for Food Security at the University of Saskatchewan. Short-read sequences were first examined with the FastQC tool [3] to evaluate sequencing quality and general metrics associated with the sequencing run. FastQC was run with default parameters. The short-read sequences were then trimmed to remove any adapter content or low quality sequences as they may lead to fragmented assemblies and erroneous base calls. Illumina©™sequencing data was filtered using Trimmomatic v0.38 [5] with filtering criteria as follows: SLIDINGWINDOW:4:28,

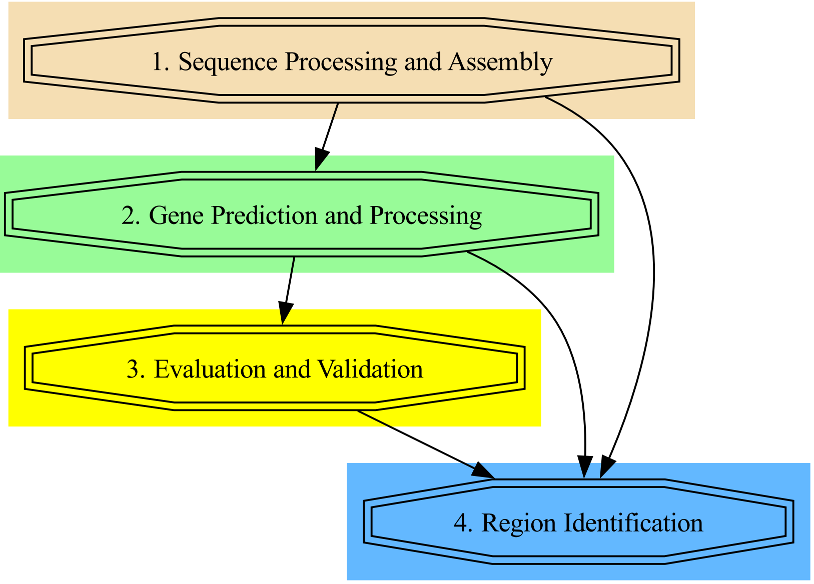

1. Sequence Processing and Assembly
2. Gene Prediction and Processing
3. Evaluation and Validation
4. Region Identification

Figure 2.1: A simplified workflow used in this work. Each processing stage is is represented by an octagon, and the flow of data are indicated by directed edges between them. In this figure and subsequent figures, the sequence processing stage is coloured brown, the gene prediction and processing stage is coloured green, the evaluation and validation stage is coloured yellow, and the region identification process is coloured blue.

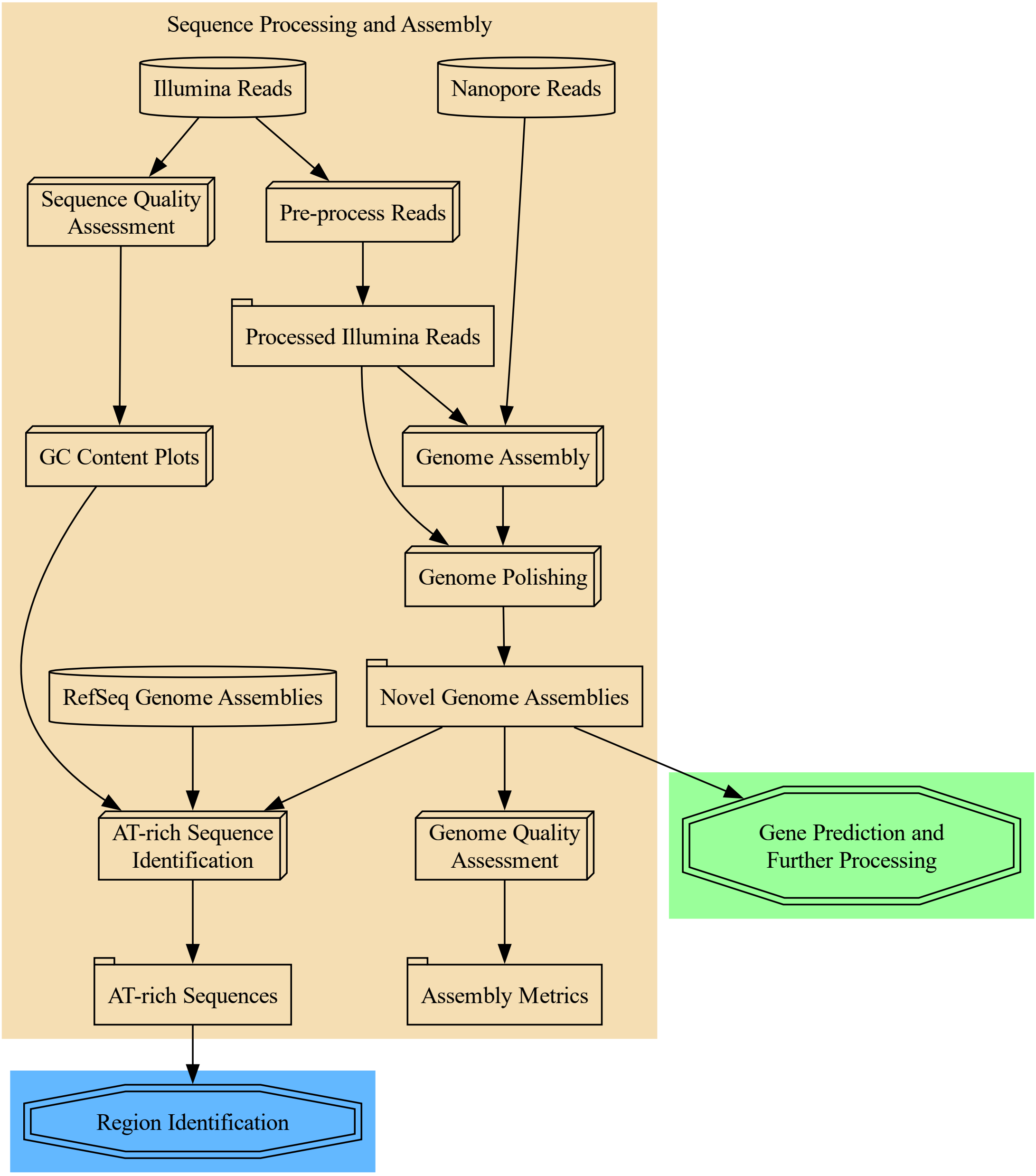


Figure 2.2: A workflow (in brown) depicting the steps taken to process input sequences, generate assemblies, and post-process those assemblies for use in other workflows. Processing steps are represented by boxes, datasets of interest that were produced by this work are represented by folders, and datasets that were not produced in this work are represented by cylinders. Other workflows are represented by green and blue arrows outside of the brown graph. Directed edges indicate the flow of data.

LEADING:28, TRAILING:28, MINLEN:75. Only surviving paired-end reads were used for further analysis. The Nanopore©™data were not processed prior to assembly.

It has been shown that assemblies that use both short-read and long-read sequencing data during the assembly are of higher quality and less fragmented than assemblies using only one form of sequencing. Genomic Nanopore©™and Illumina©™sequences from DC1 and Tsth20 were assembled in to contigs using the hybrid assembly tool NextDenovo©™ [15] v2.5.0 with default parameters. Although hybrid genome assemblies are of higher quality, they may still contain base calling errors after assembly. The highly accurate short-reads can be used to correct base calling errors after assembly. To do this, the hybrid assemblies were polished with Illumina©™sequences using NextPolish [14] v1.4.1 and default parameters. Final assembly metrics were calculated using QUAST v5.0.2 [13] with default parameters.

2.3 Overview of Gene Prediction and Further Processing

To better understand the methods used in gene prediction and further processing, the steps have been assembled into a pipeline shown in Figure 2.3. Processing steps are represented by boxes, results of interest generated by this work are represented by folders, and datasets that were not produced in this work are represented by cylinders. Other workflows that use datasets produced by this workflow are depicted as arrows outside of the green sub-graph.

First, RefSeq assemblies and their associated annotations of closely related *Trichoderma* species were selected from the National Center for Biotechnology Information (NCBI) for training and comparison. Both *ab initio* (GeneMark [6]) and evidence-based (Braker2 [7]) gene finders were selected for comparison to evaluate gene prediction behaviour of both methods in the context of *Trichoderma* genomes. Gene prediction was performed using both gene finders on the RefSeq genomes and the genomes of DC1 and Tsth20. Coding sequences output by the gene finders were then examined to better understand distributions of CDS lengths produced by different gene finders. Finally, all gene predictions were then supplied to the evaluation and validation stage as well as the region identification stage.

Datasets from the National Center for Biotechnology Information (NCBI) were selected for training and comparison, and are further discussed in Subsection 2.3.1. The GeneMark gene prediction process is described in Subsection 2.3.2, while the Braker2 gene prediction process is described in Subsection 2.3.3. The process used to examine CDSs is described in Subsection 2.3.4.

2.3.1 Selection of Datasets from NCBI

When evaluating gene-finders, it is important compare gene predictions to a standard, or control dataset, as a method of ground-truthing. Such datasets include reference genome sequences and their associated gene predictions from the NCBI as used in this work. In addition to comparing against control datasets, RNAseq data can also be used as training data for various tools as well, such as the evidence-based training used by

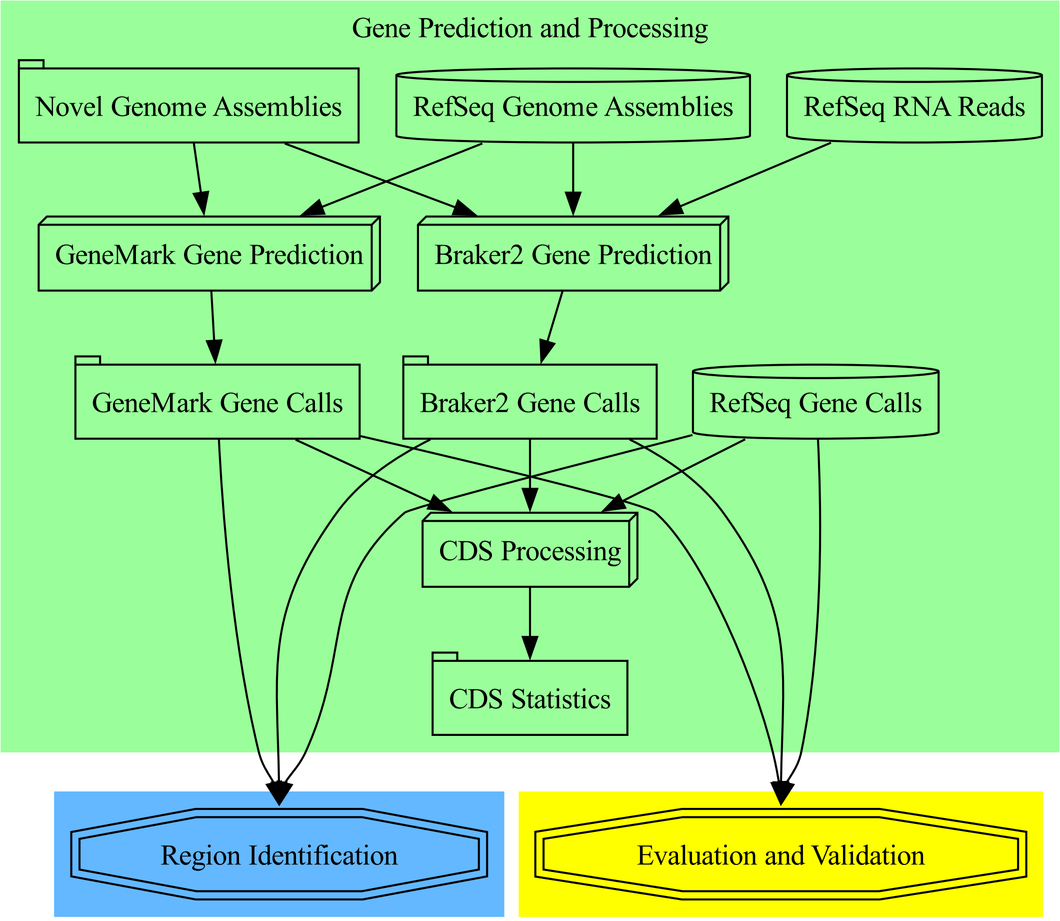


Figure 2.3: A workflow (green) depicting the steps taken to predict genes for use in other workflows and process CDS sequence lengths. Processing steps are represented by boxes, datasets of interest that were produced by this work are represented by folders, and datasets that were not produced in this work are represented by cylinders. Other workflows are represented by blue and yellow arrows outside of the green graph. Directed edges indicate the flow of data.

Braker2. In this work, four forms of these inputs were selected from NCBI for comparison and training. The first two datasets selected were assemblies and associated gene predictions of three RefSeq *Trichoderma* organisms, with those organisms being *T. virens*, *T. harzianum*, and *T. virens*. NCBI RefSeq accession IDs for those assemblies are GCF_000167675.1_v2.0, GCF_003025095.1_Triha_v1.0, and GCF_000170995.1_TRIVI_v2.0, respectively. *T. reesei* was selected as it is the most well-studied *Trichoderma* species, and widely considered to be the gold-standard for *Trichoderma* genomes [12]. *T. harzianum* and *T. virens* were selected as additional datasets due to their prevalence in literature as well.

The second set of selected inputs were RNAseq datasets from *T. reesei*, used as input in the training process for Braker2. The use of training data in gene finding applications is to produce a predictive model that factors in experimental evidence for a 'tailored' gene prediction model. These models can predict genes with more confidence and predict the correct gene structure more often [17]. The SRA accession IDs for sequences used are: SRR5229930, SRR8329344, SRR8329345, SRR8329346, and SRR8329347. Limited RNAseq datasets were used as there were few RNAseq datasets available at the beginning of this project.

The final set of inputs are protein sequences from RefSeq individuals *T. atroviride*, *Fusarium graminarium*, and *Saccharomyces cerevisiae*, which were for downstream validation of gene predictions. The NCBI accession IDs for these assemblies and associated protein sequences are GCF_000171015.1, GCF_000240135.3, and GCF_000146045.2, respectively.

2.3.2 Gene Prediction Using GeneMark-ES

Ab initio gene finding was performed using GeneMark-ES v4.71 [6] with DC1, Tsth20 and selected RefSeq *Trichoderma* assemblies used as input. Default parameters were used in all cases except for the fungal option, which allowed the use of a GeneMark model specific to fungal genomes.

2.3.3 Gene Prediction Using Braker2

For evidence-based gene finding, Braker2 v3.0.2 [7] was used with *Trichoderma reesei* selected as the reference organism for training. Illumina©™ [4] RNAseq datasets from *T. reesei* were downloaded from the NCBI short-read archive using the sra-toolkit [1] and were not trimmed or filtered prior to their use in Braker2 training. Default parameters were used to train a Braker2 model on *Trichoderma reesei* data except in the case of the fungal option, which was used for this analysis for improved gene-finding performance in fungi. The trained Braker2 *T. reesei* prediction model was then applied to all assemblies with default parameters.

Braker2 also provides a UTR option to predict upstream sequence features, but that option is experimental and was left off for this work.

2.3.4 Statistical Analysis of CDS Lengths

The distribution of gene lengths predicted by a gene-finder is an important feature to consider when selecting a gene finding tool, as genes of a particular length may be of interest to researchers, but more importantly, gene-finders may be biased to certain distributions based on their underlying models. This is important as the distribution of predicted gene lengths may also play a role in *Trichoderma's* role as an avirulent plant symbiont. These biased distributions may differ from the true distribution of CDS lengths leading to propagation of sub-optimal gene-finding results and results that may not reflect the true distribution of gene lengths. To determine, if differences exist between distributions of predicted gene lengths from each tool, the $\log_{10}$ lengths of coding sequences (CDS) were subject to a two-sample Kolmogorov-Smirnov test [2].

2.4 Overview of Evaluation and Validation of Gene Predictions

Following the gene prediction process, prediction results can then be evaluated and queried against exiting datasets as a form of validation. The workflow for evaluation and validation of gene predictions is shown in Figure 2.4. Processing steps are represented by boxes, results of interest generated by this work are represented by folders, and datasets that were not produced in this work are represented by cylinders. Other workflows that use datasets produced by this workflow are depicted as arrows outside of the yellow sub-graph. The flow of data is represented by directed edges.

Coding sequences from the previous stage were evaluated and validated using several tools. Firstly, protein products from predicted CDSs were processed with InterProScan to identify Pfam domains as evidence for the validity of gene predictions. Secondly, CDSs were analyzed with BUSCO to assess prediction of conserved fungal genes. Finally, RefSeq proteins from related fungal species were used in tblastn searches of the five *Trichoderma* assemblies selected for the main body of this work.

BUSCO evaluation, InterProScan Annotation, and BLAST alignments processes are described in more detail in Sections 2.4.1, 2.4.2, and 2.4.3, respectively. Results from these processes are also used as inputs in the region identification workflow later.

2.4.1 Evaluating Gene-Finder Performance Using BUSCO

While novel gene predictions are of great interest in new subjects, it is also important to confirm that gene-finders are predicting genes that are expected to be present in the organism of interest [18]. To determine if this is true, BUSCO v5.7.1 [21] was selected as a tool to determine a gene-finder's ability to predict a set of conserved ortholagous genes. The BUSCO Metaeuk pipeline was run using the BUSCO fungi_Odb10 fungal lineage dataset to capture highly conserved genes in fungal genomes. This pipeline was applied to all genome assemblies used in this work.

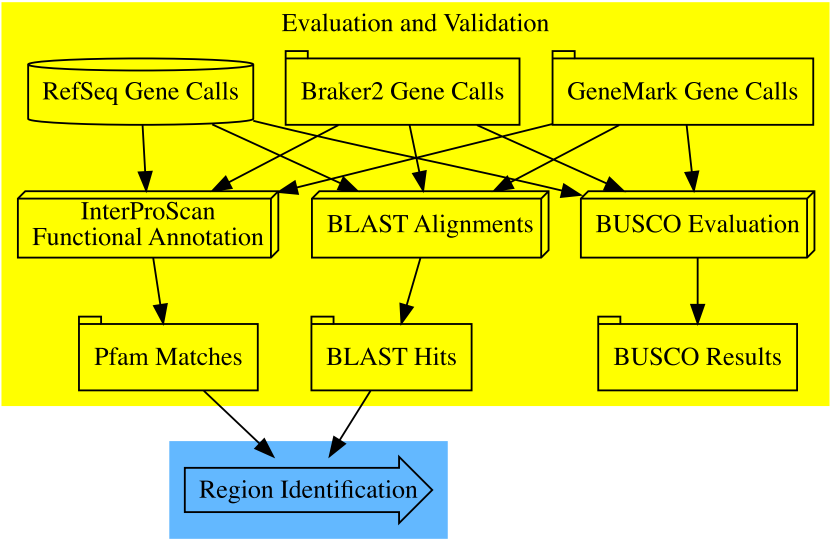


Figure 2.4: A workflow (yellow) depicting the steps taken to evaluate and validate predicted genes. Processing steps are represented by boxes, datasets of interest that were produced by this work are represented by folders, and datasets that were not produced in this work are represented by cylinders. Other workflows using results from these processing steps are represented by a blue arrow outside of the yellow graph. Directed edges indicate the flow of data.

2.4.2 Functional Annotation Using InterProScan

The presence of a functional component in a predicted gene's protein product is good evidence in favour of a gene-finder's performance. To identify functional components, the protein products from each gene finder's predictions were functionally annotated using InterProScan v5.65-97.0 [16] without the match lookup service and with the following analyses included: AntiFam-7.0, CDD-3.20, Coils-2.2.1, FunFam-4.3.0, Gene3D-4.3.0, Hamap-2023_01, MobiDBLite-2.0, NCBIfam-13.0, PANTHER-18.0, Pfam-36.0, PIRSF-3.10, PIRSR-2023_05, PRINTS-42.0, ProSitePatterns-2022_05, ProSiteProfiles-2022_05, SFLD-4, SMART-9.0, SUPERFAMILY-1.75. InterProScan was run on all predicted proteins using default parameters. The resulting protein annotations were mapped back to their genome assemblies and processed using the region identification algorithm in section 2.5.1.

2.4.3 Ground-truthing with the Basic Local Alignment Search Tool (BLAST)

The *Trichoderma* assemblies chosen for comparison were used as the subject sequences in tblastn [11] searches with the query proteins discussed in paragraph three of Section 2.3.1. Default parameters were used in the tblastn searches. Hits from the tblastn alignments were then filtered for alignments with a minimum of 30% query coverage and 30% identity. These criteria were selected as they filter out spurious alignments while also retaining short alignments that capture functional or conserved protein coding sequence. Remaining alignments were then processed using the region identification algorithm in 2.5.1

2.5 Region Identification Overview

A major portion of this work was the process of identifying regions of overlapping features as a method of comparing gene calls from different assemblers. The definition of a region and the algorithm used to identify them is described in detail in Section 2.5.1, and an overview of the region identification workflow is shown in Figure 2.5. The top of the graph shows the gene calls produced by the workflow described in Section 2.3, which are used as the main input to the region identification process along with other previously generated datasets. These additional datasets may contain other information such as blast alignments or functional annotations and can be included as features in the region identification process. Datasets outlined with rectangles indicate that the data within were processed in the context of the main gene calls and are discussed in their own results sections. Datasets output by the region identification process were then visualized with the Integrative Genomics Viewer, described in Section 2.5.3

2.5.1 Region Identification Algorithm

Predictions and results from 2.3.2 and 2.3.3 were processed into regions of overlapping gene predictions and annotations. A region is defined as a pair of genomic coordinates containing one start and one stop coordinate,

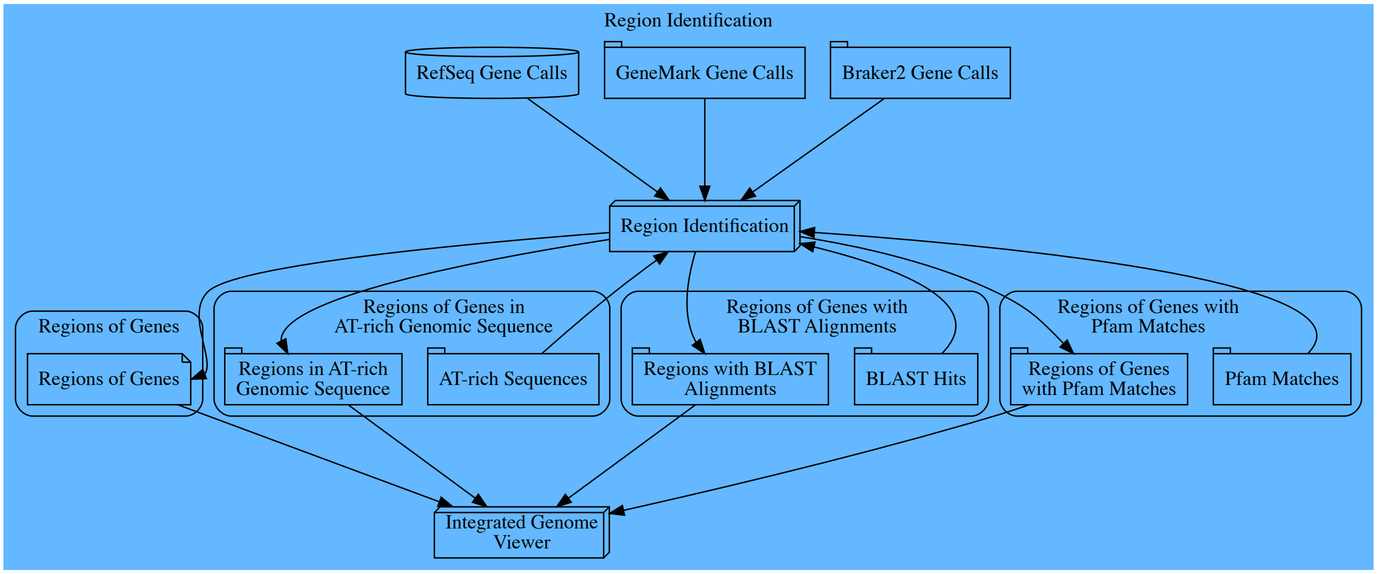


Figure 2.5: An overview of the region identification process for different datasets. Sections of the graph are outlined with rectangles, indicating that the datasets within were processed along with gene calls and discussed in their own results sections. Processing steps are represented by boxes, datasets of interest that were produced by this work are represented by folders, and datasets that were not produced by this work are represented by cylinders. Directed edges indicate the flow of data.

that contains one or more overlapping features. We define a feature as a pair genomic coordinates containing one start and one stop coordinate that associated with a gene prediction, annotation, or other data point of interest.

Overlapping features were processed into regions using a concept similar to Allen's interval algebra [9]. The algorithm first sorts all features on each contig by start coordinate and then iterates over each feature, identifying overlaps with previous features to identify regions of overlapping features. We approached this as a graph problem, where features make up a set of nodes V, and spatial overlaps between features make up a set of edges E, forming the graph $G = \{V, E\}$. This algorithm performs a transitive closure R^+ of G, resulting in sub-graphs, or regions of features in which each node has a path to each other node in that sub-graph. Pseudo-code for the algorithm is shown in Algorithm 1. This processing was performed with Python v3.12.4 [10] and the GFF package from BCBio [8] for easy GFF parsing. The resulting regions were then further processed to identify trends and behaviours of gene finders in the context of other annotated information and features.

2.5.2 Gene predictions in AT-rich Genomic Sequence

Initial exploration of the Illumina sequence data generated for DC1 and Tsth20 showed bi-modal distributions of GC content in the sequences. An example of the GC distribution of Illumina sequences for DC1 is shown in Figure 2.6 compared to a theoretical normal distribution. To differentiate AT-rich sequences from normal sequences, a cutoff value was selected to isolate AT-rich sequences from the rest. The cutoff value was chosen as 28% GC content, which is roughly the point where the AT-rich sequence counts begin increasing in all assemblies. The details of this process are explained below.

AT-rich genomic sequences were identified using sliding windows of GC content over each *Trichoderma* assembly. Sliding windows were generated using the isochore program from the EMBOSS tool suite v6.6.0 [19]. The sliding window used was 250bp wide with a 50bp shift. A window was classified as AT-rich if the sequence contained lower than 28% GC content. The genomic sequences from AT-rich windows were mapped to a GFF file and processed using the region identification algorithm described in section 2.5.1.

2.5.3 Visualization of Identified Regions

Results from the region identification process were visualized using the IGV genome browser v2.19.1 [20]. An example of an IGV screenshot is shown in Figure 2.7.

Algorithm 1 The general algorithm underlying the region identification process.

```
INPUT: list of GFF files, $GFFList$
RETURNS: list of regions, $regionList$
for <each GFF in GFFList> do
    $allFeatures \leftarrow GFF.features$
end for
$sortedFeatures \leftarrow sortStartCoord(allFeatures)$
$regionList \leftarrow []$
$firstFeature \leftarrow firstItem(sortedFeatures)$
$lastFeature \leftarrow lastItem(sortedFeatures)$
for <each feature in sortedFeatures> do
    if $feature = firstFeature$ then
        $currentRegion \leftarrow newRegion(feature)$
        $currentRegion.start \leftarrow feature.start$
        $currentRegion.end \leftarrow feature.end$
    else if $feature = lastFeature$ then
        $regionList.append(currentRegion)$
        $return\ regionList$
    else if $feature\ overlaps\ currentRegion$ then
        $currentRegion.updatePositions(currentRegion, feature)$
        $currentRegion.addFeature(feature)$
    else
        $regionList.append(currentRegion)$
        $currentRegion \leftarrow newRegion(feature)$
        $currentRegion \leftarrow feature.start$
        $currentRegion \leftarrow feature.end$
    end if
end for
$return\ regionList$
```

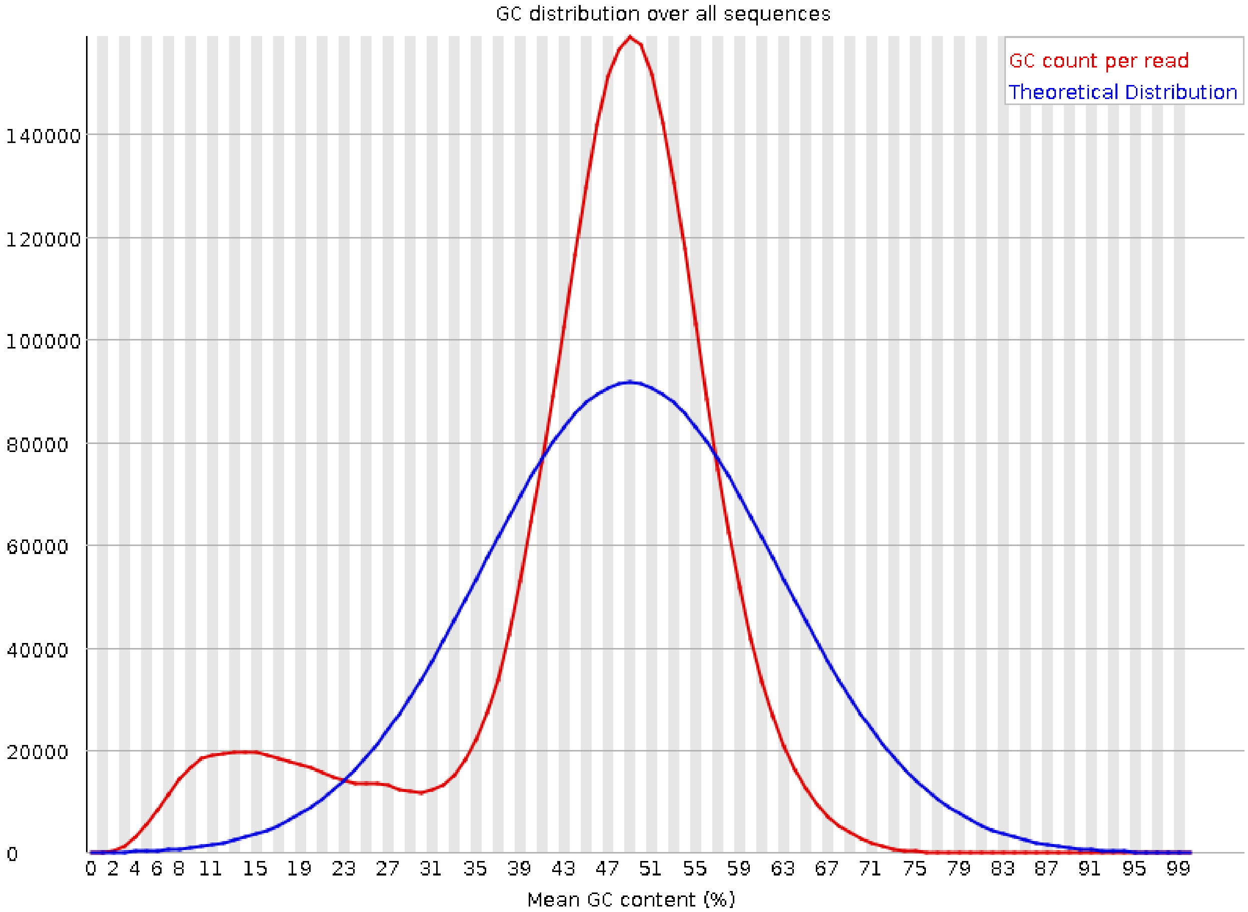
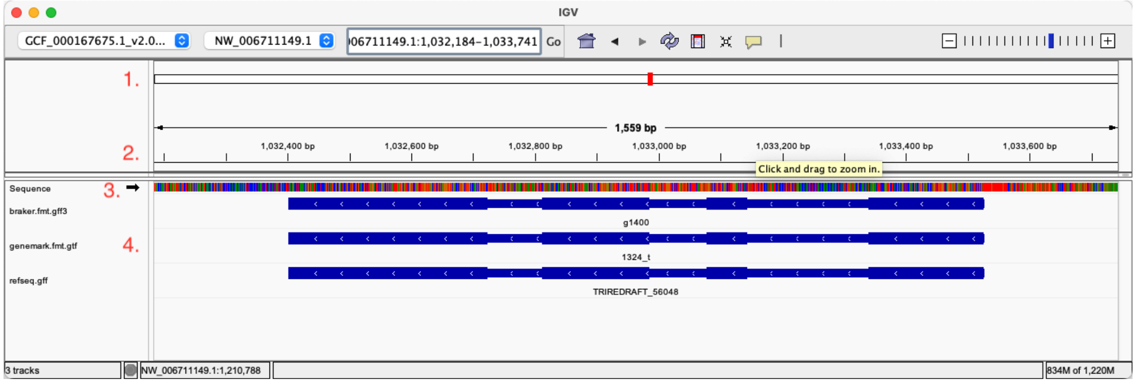


Figure 2.6: A plot of GC content of Illumina sequences generated for DC1. The X-axis indicates mean GC content of sequences while the Y-axis indicates the total number of sequences for a given mean GC content. A theoretical normal distribution is overlaid in blue.



GCF_000167675.1_v2.0...

NW_006711149.1

)06711149.1:1,032,184-1,033,741

Figure 2.7: An example of an IGV screenshot used in portions of the results from this work. The top half of the image displays information about the reference sequence with track one showing the position of the viewer relative to the reference sequence, and track two showing the numerical positions of the reference. The lower half of the image displays more tracks in the form of additional layers of information mapped to the reference sequence. Track three shows a track displaying the nucleotide at each position if the scale of the viewer is small enough. Track four comprises several tracks, each containing features, the regions in blue, from a GFF file. The tracks in this example are limited to gene predictions, although tracks in other IGV screenshots may include features from other tools such as InterProScan and will be discussed in their respective sections. Individual tracks will be referred to in increasing numerical order in the track list beginning from the top. The nucleotide track is not included in the order.

References

[1] SRA-Toolkit. https://hpc.nih.gov/apps/sratoolkit.html.

[2] Kolmogorov–Smirnov Test. In *The Concise Encyclopedia of Statistics*, pages 283–287. Springer New York, 2008.

[3] Simon Andrews. Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence Data. https://www.bioinformatics.babraham.ac.uk/projects/fastqc/.

[4] Simon Bennett. Solexa Ltd. *Pharmacogenomics*, 5(4):433–438, June 2004.

[5] Anthony M. Bolger, Marc Lohse, and Bjoern Usadel. Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics*, 30(15):2114–2120, August 2014.

[6] Mark Borodovsky and Alex Lomsadze. Eukaryotic gene prediction using GeneMark.hmm-E and GeneMark-ES. *Current Protocols in Bioinformatics*, Chapter 4(SUPPL.35):4.6.1–4.6.10, September 2011.

[7] 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, March 2021.

[8] Brad Chapman. BCBio Python Package.

[9] Rina Dechter. Chapter 12 - Temporal Constraint Networks. In Rina Dechter, editor, *Constraint Processing*, pages 333–362. Morgan Kaufmann, 2003.

[10] Python Software Foundation. Python.

[11] E. Michael Gertz, Yi-Kuo Yu, Richa Agarwala, Alejandro A. Schäffer, and Stephen F. Altschul. Composition-based statistics and translated nucleotide searches: Improving the TBLASTN module of BLAST. *BMC Biology*, 4(1):41, December 2006.

[12] Vijai Kumar Gupta, Andrei Stecca Steindorff, Renato Graciano de Paula, Rafael Silva-Rocha, Astrid R. Mach-Aigner, Robert L. Mach, and Roberto N. Silva. The Post-genomic Era of Trichoderma reesei: What's Next? *Trends in Biotechnology*, 34(12):970–982, December 2016.

[13] Alexey Gurevich, Vladislav Saveliev, Nikolay Vyahhi, and Glenn Tesler. QUAST: Quality assessment tool for genome assemblies. *Bioinformatics (Oxford, England)*, 29(8):1072–1075, April 2013.

[14] Jiang Hu, Junpeng Fan, Zongyi Sun, and Shanlin Liu. NextPolish: A fast and efficient genome polishing tool for long-read assembly. *Bioinformatics*, 36(7):2253–2255, April 2020.

[15] Jiang Hu, Zhuo Wang, Zongyi Sun, Benxia Hu, Adeola Oluwakemi Ayoola, Fan Liang, Jingjing Li, José R. Sandoval, David N. Cooper, Kai Ye, Jue Ruan, Chuan-Le Xiao, Depeng Wang, Dong-Dong Wu, and Sheng Wang. NextDenovo: An efficient error correction and accurate assembly tool for noisy long reads. *Genome Biology*, 25(1):107, 2024.

[16] Philip Jones, David Binns, Hsin-Yu Chang, Matthew Fraser, Weizhong Li, Craig McAnulla, Hamish McWilliam, John Maslen, Alex Mitchell, Gift Nuka, Sebastien Pesseat, Antony F. Quinn, Amaia Sangrador-Vegas, Maxim Scheremetjew, Siew-Yit Yong, Rodrigo Lopez, and Sarah Hunter. InterProScan 5: Genome-scale protein function classification. *Bioinformatics*, 30(9):1236–1240, 2014.

[17] Chengzhi Liang, Long Mao, Doreen Ware, and Lincoln Stein. Evidence-based gene predictions in plant genomes. *Genome Research*, 19(10):1912–1923, October 2009.

[18] Mosè Manni, Matthew R. Berkeley, Mathieu Seppey, and Evgeny M. Zdobnov. BUSCO: Assessing Genomic Data Quality and Beyond. *Current Protocols*, 1(12):e323, 2021.

[19] P. Rice, I. Longden, and A. Bleasby. EMBOSS: The European Molecular Biology Open Software Suite. *Trends in genetics : TIG*, 16(6):276–277, June 2000.

[20] James T. Robinson, Helga Thorvaldsdóttir, Wendy Winckler, Mitchell Guttman, Eric S. Lander, Gad Getz, and Jill P. Mesirov. Integrative genomics viewer. *Nature Biotechnology*, 29(1):24–26, 2011.

[21] Felipe A. Simão, Robert M. Waterhouse, Panagiotis Ioannidis, Evgenia V. Kriventseva, and Evgeny M. Zdobnov. BUSCO: Assessing genome assembly and annotation completeness with single-copy orthologs. *Bioinformatics*, 31(19):3210–3212, October 2015.

[22] Yunhao Wang, Yue Zhao, Audrey Bollas, Yuru Wang, and Kin Fai Au. Nanopore sequencing technology, bioinformatics and applications. *Nature Biotechnology*, 39(11):1348–1365, 2021.