Spring 2020

Efficient comparative genomics with low coverage data using PALADIN

Rachel Cates
University of New Hampshire, Durham

EFFICIENT COMPARATIVE GENOMICS WITH LOW COVERAGE DATA USING PALADIN

BY

RACHEL CATES

THESIS

Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Master of Science
in
Computer Science

December. 2019

This thesis was examined and approved in partial fulfillment of the requirements for the degree of Master of Science in Computer Science by:

Thesis director, Dr. R. Daniel Bergeron, Professor Emeritus of Computer Science

Dr. W. Kelley Thomas, Professor of Molecular, Cellular, and Biomedical Sciences

Anthony Westbrook, Computational Scientist

On December 10th 2019

Approval signatures are on file with the University of New Hampshire Graduate School.

I wish to extend thanks to all those involved in the research and development of the *PB&J tool*, including: Dr. R. Daniel Bergeron, Anthony Westbrook, and Dr. W. Thomas Kelley at the University of New Hampshire, as well as members of the original research group: Sai Cummings, Matt Strobel, and Sidney Birch. I also wish to thank those at the Hubbard Center for Genome Studies for providing access to the Premise and Ron servers which were used for all development of this tool.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

EFFICIENT COMPARATIVE GENOMICS WITH LOW COVERAGE DATA (USING PALADIN)

by

Rachel Cates

Comparative genomics focuses on comparing the catalog of genomic elements of whole genome sequences to determine the functional relationship between genes. One of the first steps in comparative analysis is to make genome assemblies for the organisms of interest. However, due to the cost and time required to create these assemblies, only a limited number of organisms can be analyzed simultaneously. A new software package, PALADIN, maps nucleotide reads in protein space using a modified version of the Burrows-Wheeler Aligner (BWA). We demonstrate that PALADIN can accurately and efficiently identify the proteins in a genome using inputs with significantly lower coverage levels than traditional comparative genomics methods. Moreover, we provide an easy to use PALADIN plugin, *PB&J*, that simplifies the use of PALADIN for comparative genomics.

CHAPTER 1

INTRODUCTION

Comparative genomics focuses on comparing the catalog of genomic elements of whole genome sequences from multiple related organisms to determine the functional relationship between genes. The traditional work flow involves generating a genome assembly for each organism, which includes sequencing DNA, performing an assembly, and creating an annotation. Once the assembly is complete, the annotation output can be used with numerous software packages to perform comparative analyses. Unfortunately, this approach is barely usable because sequencing and assembly are very costly and time consuming. This is partly due to the amount of sequencing that must be completed in order to generate an accurate assembly [1]. Depending on the type of analysis being performed, the recommended coverage for Illumina sequencing is between 10-30x coverage, where coverage is the average number of reads covering a given base in the genome [2].

A new software package, PALADIN, developed by Westbrook et al. [3] aligns DNA reads in fasta format to protein references from the UniProt database. It does this by modifying and extending the popular Burrows-Wheeler Aligner (BWA) mapping tool [4] to align nucleotides in protein space. The aim of this research is to verify that PALADIN can be used as an alternative approach for comparative genomic analysis. Unlike traditional comparative genomics pipelines, PALADIN captures a high quality representation of a sample's protein content at read coverage levels that are a fraction of what traditional methods require. Not only is this low coverage approach significantly more cost effective, but it also bypasses the genome assembly step, which has its own set of complexities that can introduce error into a study.

This approach prompts the two-part question: What is the minimum amount of coverage needed to accurately annotate a genome, and how do we define "accurately"? It is not feasible to do a comprehensive direct comparison between PALADIN results and genome assembly results because of the extremely high cost of assembly. Instead, we show that PALADIN results compare favorably with those obtained using the BUSCO comparative genomics tool [5].

BUSCO (Benchmarking Universal Single-Copy Orthologs) is often used to measure the quality of an annotation based on how close it comes to hitting all genes present. BUSCO is a software library and a set of high quality reference species datasets. It provides quantitative information on the completeness of a given genome based on the number of genes known or expected to be present in it. These datasets are frequently updated, and BUSCO is widely considered by those in the genomics field to be an essential tool for comparative genomics.

To assess the accuracy and completeness of PALADIN for the identification of BUSCO genes using low coverage data, we developed the PALADIN-BUSCO-Join ("PB&J") tool. To validate its effectiveness, we fed to PALADIN known genomes whose BUSCO genes have been identified. The PB&J tool was then used to compare PALADIN's gene presence identification output with BUSCO gene output for each genome.

CHAPTER 2

METHODS

2.1 Pipeline overview

The *PB&J* tool is written in Python3 with Python's standard library. It has several parameter options that together form a pipeline (Fig. 1). First, the *map* command creates a data structure that converts between the KBIDs used by PALADIN and the OrthoIds used by BUSCO. This data structure is written to a file called the "OrthoKBID cross-reference file". This file is used with the *score* command, which determines the number of complete, fragmented, duplicate, and missing BUSCO genes correctly detected by PALADIN, thereby giving the user a measure of genome completeness. If the number of correctly detected BUSCO genes is close to the total BUSCO genes for that species, the user can be confident that their genome is relatively complete and can then use the tool's *compare* command, which generates presence-absence data for any number of PALADIN-produced .tsv files. Each .tsv file contains all the genes present in that species. Therefore, the presence-absence data produced from the *compare* command includes matrix and list views indicating the genes shared across each species for which a .tsv file is provided.

2.2 Genomes used

The primary data for this research consists of whole genomes from several species assessed in the original BUSCO paper including representatives from *nematoda*, *fungi*, and *gammapro-*

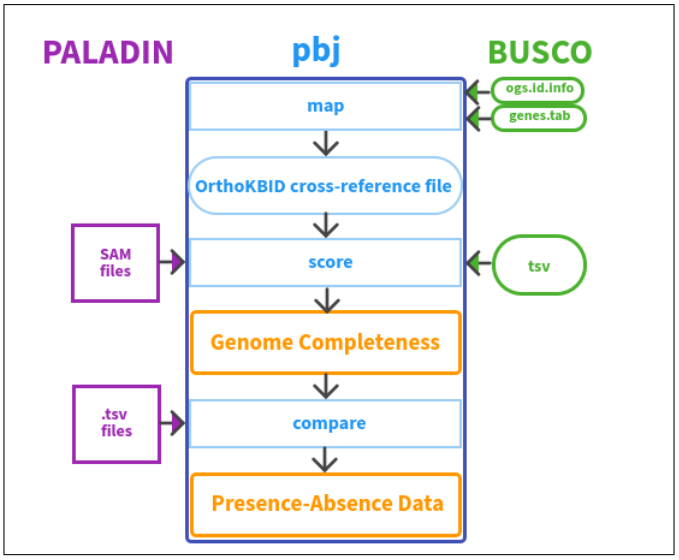


Figure 1: Pipeline that comprises the *PB&J* tool. In step one a BUSCO compatible data structure for PALADIN is created and written to a file using the *map* command. This data structure is used with the *score* command to help assess the genome completeness of species from PALADIN alignments as compared to BUSCO. If species are found to have high genome completeness from the PALADIN alignments, the user has the option to run the *compare* command in order to generate a presence-absence matrix for them.

teobacteria (see appendix A). The selected genomes were all run through the ART bioinformatics toolkit [6] to generate single end 250 base-pair Illumina reads at 1x, 3x, 5x, 10x, 20x and 50x average coverages. We used single-end reads since paired-end reads are not currently supported by PALADIN. The resulting FastQ and SAM files were used in all subsequent research.

2.3 Generating the control data

2.3.1 OrthoKBID cross-reference files

PALADIN and BUSCO use different gene identification conventions, which presents problems when comparing their outputs. PALADIN uses KBIDs from the UniProt database [3], whereas BUSCO uses OrthoIds from OrthoDB [5]. Although PALADIN produces a data file that converts between KBIDs and the ids used by a variety of other databases, the file does not contain OrthoIds. Consequently we need an extra step of cross referencing with BUSCO's *ogs.id.info*

file for each species of interest.

PB&J's *map* command addresses this issue by creating the OrthoKBID cross-reference file which consists of two tab-separated columns that contain OrthoIds and their corresponding KBIDs. This file can be efficiently read and stored as a hash table to validate the results later produced by PALADIN. Due to the species used in this research (see appendix A), we created OrthoKBID cross-reference files at a UniProt 90 cluster level for *fungi*, *nematoda*, *gammaproteobacteria*, *actinobacteria*, *bacteroidetes*, *cyanobacteria* and *firmicutes*.

2.3.2 Parameter tuning

The *PB&J* tool includes a range of parameter options for the *score* command. In order to quantify the usefulness of these parameters for increasing BUSCO gene detection given low coverage data, we created two supplementary tools that together form an additional pipeline. The *PBtracker* tool is the first in this additional pipeline, and uses the OrthoKBID cross-reference file along with FastQ and SAM files produced by the ART Illumnina runs to cross reference both EMBL and non-EMBL ids to their corresponding BUSCO ids. These BUSCO ids, along with the absolute and relative start and stop positions of their corresponding read, are added to their appropriate FastQ headers, producing a modified FastQ file. The modified FastQ file is used by PALADIN, whose output is assessed by *PBtester*, the second supplementary tool. As described below, we performed a series of *PBtester* runs using all combinations of multiple parameter values for alignment length, mapping quality, category scores, suboptimal hit detection, alignment scores, and chimeric hit processing.

Alignment length

In order to discard low quality data, *PBtester* allows us to apply a minimum alignment length filter to each SAM file. For each species at each coverage level, *PBtester* was run with the following minimum alignment length filters: 0, 100, 150, 200, 220, and 240. Any data with an alignment length below the specified minimum alignment length was not considered in that

execution of *PBtester* and in the subsequent results.

Mapping quality

We can specify a minimum mapping quality for input data to further filter out low quality data that might hurt confidence in our results. We applied each of the following minimum quality filters to each species at each coverage level and minimum alignment length: 0, 20, 30, 40. Any data with a quality score below the specified minimum quality was not considered in that execution of *PBtester* and in the subsequent results.

Category scores

Another feature of the *PBtester* tool is category score filtering. Each alignment from PALADIN falls into a category, such as "true positive" or "false negative". We wanted to account for reads that barely aligned in order to remove them from our results as they were inflating our false positive and false negative gene counts. For instance, the case where only 5% of a read aligned should not be given equal weight to the case where 80% of a read aligned. If both of these cases fell into a false positive category, our false positive count would increase by two even though one of these alignments was negligible. Therefore, we applied minimum category score filters of 20%, 40%, 50%, 60%, and 70% to our existing parameter settings.

Secondary hit detection

When this flag is turned on *PBtester* considers secondary, also called suboptimal, hits as well as primary hits from the SAM file. Results are not only reported for primary hits alone, but also for secondary hits and for aggregated primary and secondary hits. Since it provides additional data, this flag was turned on for all of our *PBtester* executions.

Alignment scores

Considering aggregated primary and secondary hits presents the problem of determining the precise secondary hits to include, i.e., those that help BUSCO gene detection and do not hurt final scores by introducing false positives and false negatives. One criteria we assessed to this end was the difference between alignment scores of each secondary hit and all of its associated primary hits. Results were normalized by dividing by the alignment's length, which was deduced from the length of the alignment's string in the corresponding line of the SAM file. Similarly, we considered the ratio between the primary and secondary alignment score as another option for determining when to include a particular secondary hit.

Chimeric hit processing

PALADIN has the ability to identify chimeric hits. Therefore, we tested whether chimeric hits were a significant component of the improvement achieved by including secondary hits. For each parameter combination we executed separate *PBtester* runs filtering only for chimeric hits, only for non-chimeric hits, and for both chimeric and non-chimeric hits.

Open reading frame detection

Not a parameter for *PBtester*, but one we also considered in order to fully optimize our pipeline, is PALADIN's open reading frame ("ORF") detection feature. PALADIN allows users to filter out reads with no ORF above a specified length. Turning this parameter on allows for decreased computation time at the cost of lower quality results. We investigated whether PALADIN can attain the high quality outputs that result from having ORF turned off and yet preserve the low computational costs of having ORF turned on. For data we used FastQ files produced by ART Illumina for *C. gattii* at 1x, 3x, 5x, 10x, 20x, and 50x average coverages. On each of these coverage levels we ran ten executions of PALADIN with ORF on, and ten more with ORF off. This process was repeated on five separate occasions to make a total of 100 runs per coverage level. Real and CPU times for all runs were recorded and saved with the UNIX *time* command.

All commands were executed as *slurm* [7] jobs on the University of New Hampshire's *premise* server, using two 12-core Intel(R) Xeon(R) CPUs @2.50GHz and 512GB of main memory.

2.3.3 PBtester outputs

For each of the parameter combinations described in section 2.3.2, the following files were produced by *PBtester*:

Basic statistics

This file includes basic statistics of the run such as mean alignment length and mean mapping quality, as well as the parameters selected for that execution.

Extended confusion matrices

To avoid conflating alignments of similar category, we developed a naming convention to characterize them in the most granular way possible (Table 1). *PBtester* puts every gene and every alignment from a given SAM file into one of eleven categories. This naming convention allows us to account for intergenic and undefined regions of a read and ultimately allows us to compute the total number of true positives, false positives, true negatives, false negatives, and mismatches detected by PALADIN. We can be confident that no alignment is missed or considered more than once.

Abbreviation derivation for the first five categories is straightforward. 'tp' stands for 'true positive', 'tn' stands for 'true negative', and 'mm' stands for 'mismatched'. Therefore, 'tpb' is short for 'true positive BUSCO', 'tpn' is short for 'true positve non-BUSCO, 'mmb' is short for 'mismatched BUSCO', and 'mmn' is short for 'mismatched non-BUSCO'.

For the false negative and false positive categories we opted to exclude the 'positive' or 'negative' portion of the label since the category a gene or alignment ends up in depends on which way around one considers the source versus the result. Therefore, everything in these categories is simply considered 'false', giving us the first character of the abbreviations. To

this, the first character of the type of gene in the gff file followed by the first character of the type of gene found by PALADIN are appended. So 'fnb' stands for 'false non-BUSCO to BUSCO', 'fbn' stands for 'false BUSCO to non-BUSCO', 'fib' stands for 'false intergenic region to BUSCO', 'fin' stands for 'false intergenic region to non-BUSCO', and so on.

For example, a gene falls into the "true positive BUSCO" category if the read contains a BUSCO gene and PALADIN aligned it to the same BUSCO gene. On the other hand, if the read contains a non-BUSCO gene and PALADIN aligned it to the same non-BUSCO gene, it falls into the "true positive non-BUSCO" category. Table 1 summarizes the eleven categories where the last two columns identify sample gene names for the case.

Category	gff	PALADIN
True positive BUSCO ("tpb")	B1	B1
True positive non-BUSCO ("tpn")	NB1	NB1
True negative ("tn")	I	N/A
Mismatched BUSCO ("mmb")	B1	B2
Mismatched non-BUSCO ("mmn")	NB1	NB2
False BUSCO to non-BUSCO ("fbn")	B1	NB1
False non-BUSCO to BUSCO ("fnb")	NB1	B1
False intergenic region to BUSCO ("fib")	I	B1
False intergenic region to non-BUSCO ("fin")	I	NB1
False BUSCO to undefined region ("fbu")	B1	N/A
False non-BUSCO to undefined region ("fnu")	NB1	N/A

Table 1: A PALADIN alignment can match a BUSCO id (B1 or B2), a non-BUSCO id (NB1 or NB2), an intergenic region (I), or an undefined region (N/A)

Overlap statistics

A record of each alignment from PALADIN is printed to this file, along with all genes that it overlaps with.

Secondary hit statistics

This file contains the normalized difference (or ratio depending on the parameter setting) in alignment scores of each secondary hit and all corresponding primary hits. Each of these data

points has a boolean flag indicating whether including the secondary hit helped or hurt results.

2.4 Generating the test data for BUSCO comparison

The *PB&J* tool's *score* command requires a BUSCO .tsv file as a benchmark for comparison to PALADIN. We performed a series of BUSCO runs on each species at 3x and 50x average coverages. We generated the assembly data needed by BUSCO using the *SPAdes* assembler [8] and the same simulated read data from ART Illumina described in section 2.2. We opted to prepare the non-control data in this manner, i.e. running ART and *SPAdes* to provide input to BUSCO, since it does not use quality of the test genome's known assembly.

Since it is simple to ignore the header in column one of any SAM file, there is no need to perform separate PALADIN runs for the test phase of research. Instead, we reused the SAM files from the control phase, this time only paying attention to data that was there before they were modified by data from *PBtracker*.

CHAPTER 3

RESULTS

The next phase of research involved running the *PB&J* tool's *score* command using the parameters that were determined to be optimal based on results of the the various executions of *PBtester* discussed in section 2.3.2. After assessing genome completeness with *PBtester's* outputs we generated presence-absence matrices using *PB&J's compare* command.

3.1 Parameters selected

As described previously, we performed hundreds of evaluation tests using all combinations of multiple values for each of many different parameters. From those tests we were able to identify a small subset of parameters that we could focus on in order to determine the effectiveness of PALADIN.

3.1.1 Alignment length, mapping quality, and category scores

Very high alignment length filters remove too much data but very low alignment length filters do not remove enough low quality data, and therefore hurt confidence in our results. Therefore, a minimum alignment length filter of 200 was selected as the final parameter option for this category, however minimum alignment lengths as low as 100 will generally produce comparable results, with the addition of a few extra alignments in the fbn and fnb categories. We leave the specific selection of this parameter to the discretion of the user, but recommend that they do not go below 100.

Similarly, with extremes of minimum and maximum mapping quality, too little or too much data is removed. We gathered final results with a minimum mapping quality of 20, and this is what we recommend to potential users as a value for this parameter.

With a minimum category score filter of 40% we found that we could reduce fbn and fnb counts by as much as 50% for some species while hardly reducing the tpb counts. Increasing the category score filter much above 40% starts to have diminishing returns as it drastically reduces tpbs for some species, specifically *C. elegans*. While fbns and fnbs are also reduced slightly with the higher filters, it is too little to justify the huge decrease in tpbs for this species.

3.1.2 Primary and secondary hits

Overall, we found that including secondary hits increases true positive BUSCO counts slightly across most species while simultaneously decreasing gene detection in the "false" categories. Therefore despite persistent high counts in the false BUSCO to non-BUSCO category even when including secondary hits, we recommend running *PB&J*'s *score* command with secondary hit detection turned on.

True positive BUSCO (tpb) matches

Figs. 2 and 3 show the number of true positive BUSCO genes detected across all of the species tested using 3x and 50x coverage data, respectively. The number of genes found in this category when using only primary hits is represented by the blue bars and is similar for both coverage levels across all species. Green bars represent the total number of BUSCO genes in the species. Since true positive BUSCO counts were low in comparison to this benchmark in some cases, secondary hits as represented by the orange bars were considered in order to mitigate the problem. Across all species, adding in secondary hits increased BUSCO gene detection slightly.

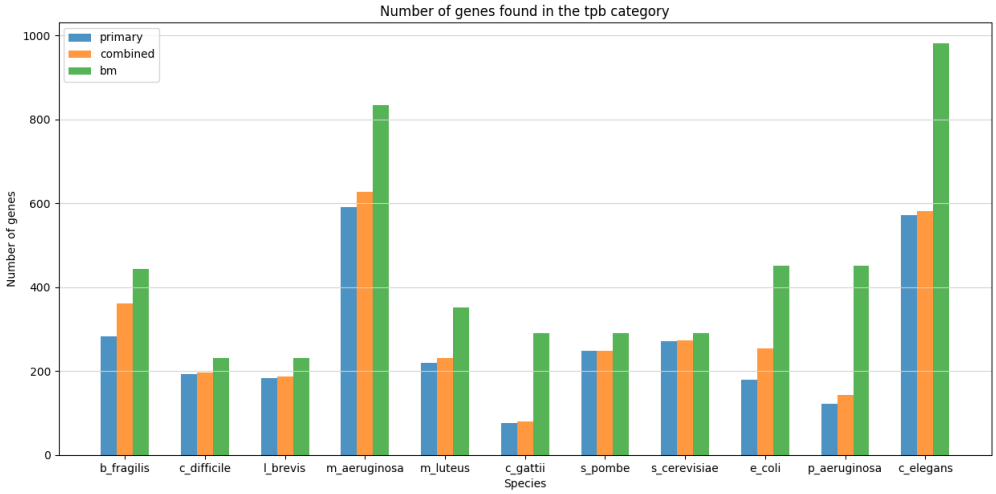


Figure 2: Number of genes detected by PALADIN in the 'tpb' category for species at 3x coverage after applying a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. Considering secondary hits slightly increased the count of true positive BUSCOs across most of the species tested.

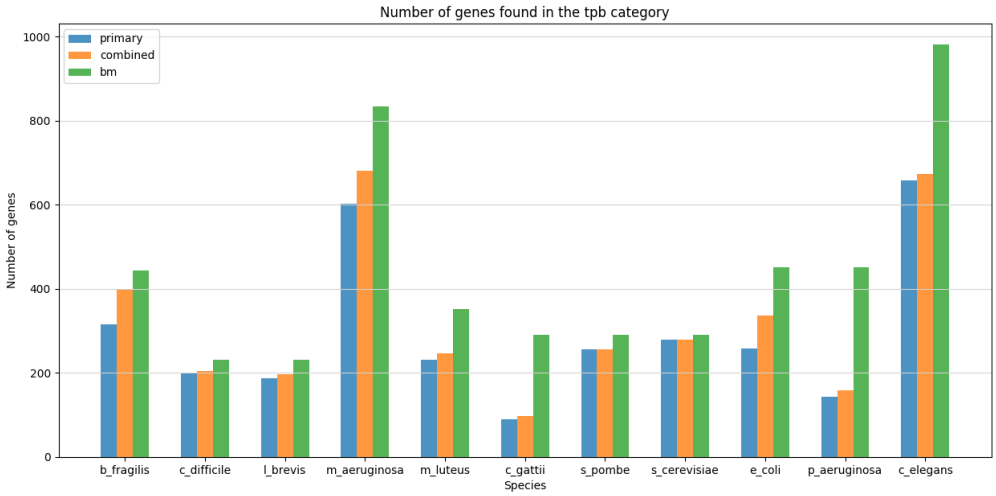


Figure 3: Number of genes detected by PALADIN in the 'tpb' category for species at 50x coverage after applying a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. Considering secondary hits slightly increased the count of true positive BUSCOs across most of the species tested.

False non-BUSCO to BUSCO (fnb) matches

Fig. 4 shows the number of false non-BUSCO to BUSCO genes across all species tested using 3x coverage data. For most of these species, adding secondary hits significantly reduces the

number of genes in this category, especially for the prokaryotes. Nevertheless, some of the results in this category are not what we had hoped for. In particular, the ratio of tpbs to fnbs for *E. coli*, *P. aeruginosa* and *C. gattii* being 255:27, 144:32 and 80:27 respectively, are higher than we would have liked. Therefore, this is an area for further validation and research.

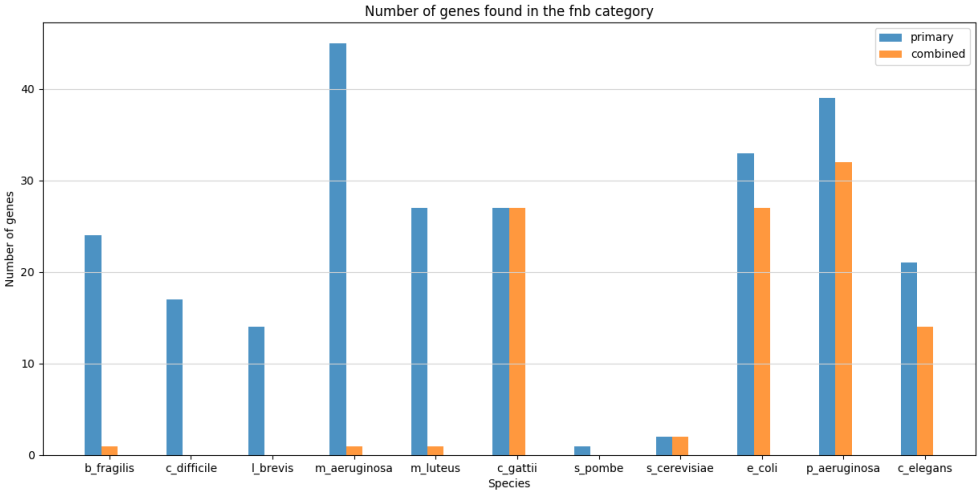


Figure 4: Number of genes detected by PALADIN in the 'fnb' category for species at 3x coverage after applying a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. Considering secondary hits decreased counts of false non-BUSCO to BUSCOs for most of the species tested.

False BUSCO to non-BUSCO (fbn) matches

The number of false BUSCO to non-BUSCO genes found is illustrated by Fig. 5. Admittedly these are the most disappointing results of the research, as counts across most species remain high despite the application of our optimal filter combination.

To determine the reason for such high counts in this category, we must first understand how many of the correct BUSCO genes for those reads did not get identified in another read. If this happens frequently, we have two problems. Firstly, we failed to identify that particular BUSCO gene and secondly we succeeded in identifying an incorrect BUSCO gene. We also need to explore whether the incorrectly mapped non-BUSCO genes mapped to the same non-BUSCO genes in other reads in the same organism. If this is frequently the case, the problem

is much less serious but could still could be an issue depending on the researcher's goals. In short, more tests are needed to uncover the reason for such high counts in this category, and therefore, this is an area for future work.

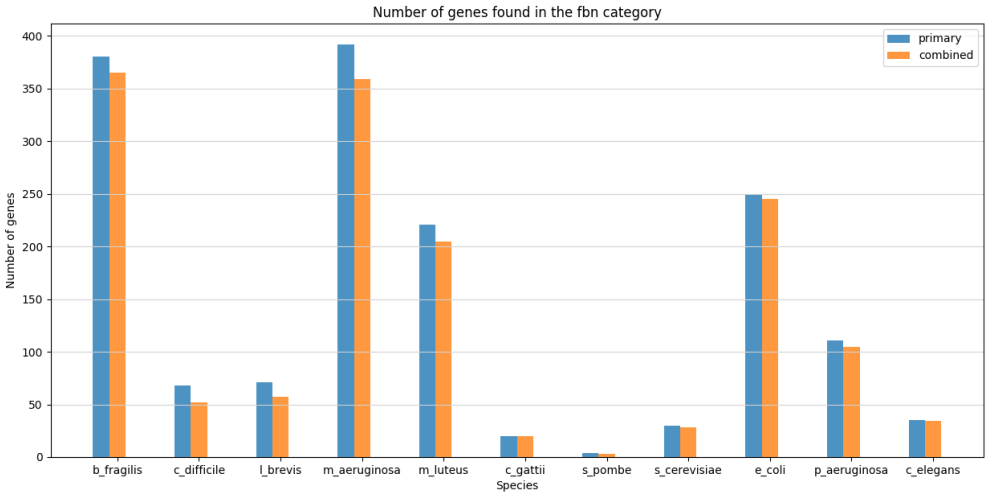
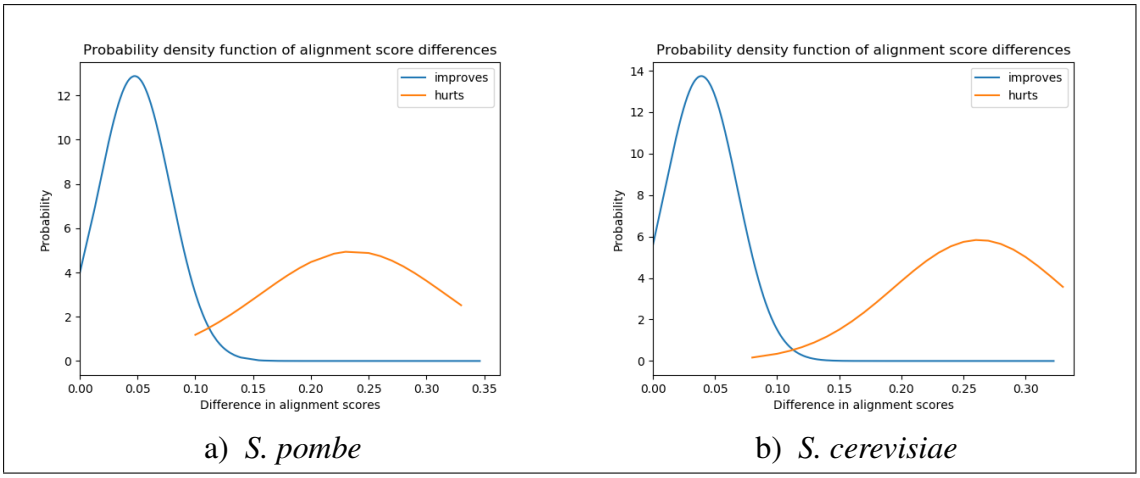


Figure 5: Number of genes detected by PALADIN in the 'fbn' category for species at 3x coverage after applying a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. Considering secondary hits decreased counts of false BUSCO to non-BUSCOs slightly for most of the species tested.

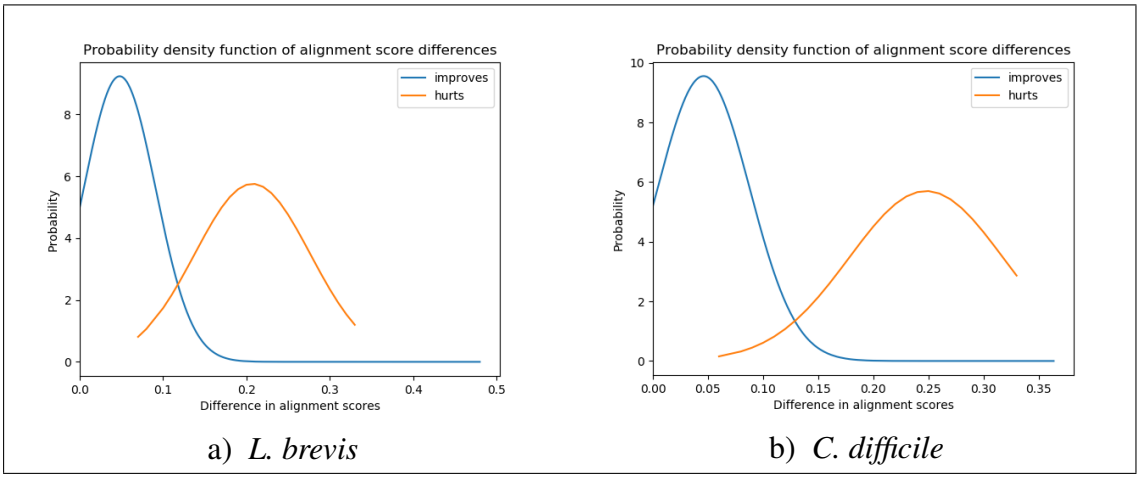
3.1.3 Alignment score differences

Figs. 6, 7, and 8 show the probability density function in normalized alignment score differences between primary and secondary hits. In other words, they show the density of the distribution of the data around a certain point on the x-axis, not the pure probability. Results are grouped by phylogenetically similar species. These plots demonstrate that taking the difference in alignment score between primary and secondary hits is a promising heuristic for predicting when secondary hits help and hurt results. In particular, when the normalized difference in alignment score is between 0.01 and 0.05, there is a high probability across all the species tested that the secondary hit in question will improve the overall results. We still need to test more organisms to solidify phylogeny-based recommendations for this parameter, so for now we opt to stick with this general guideline.



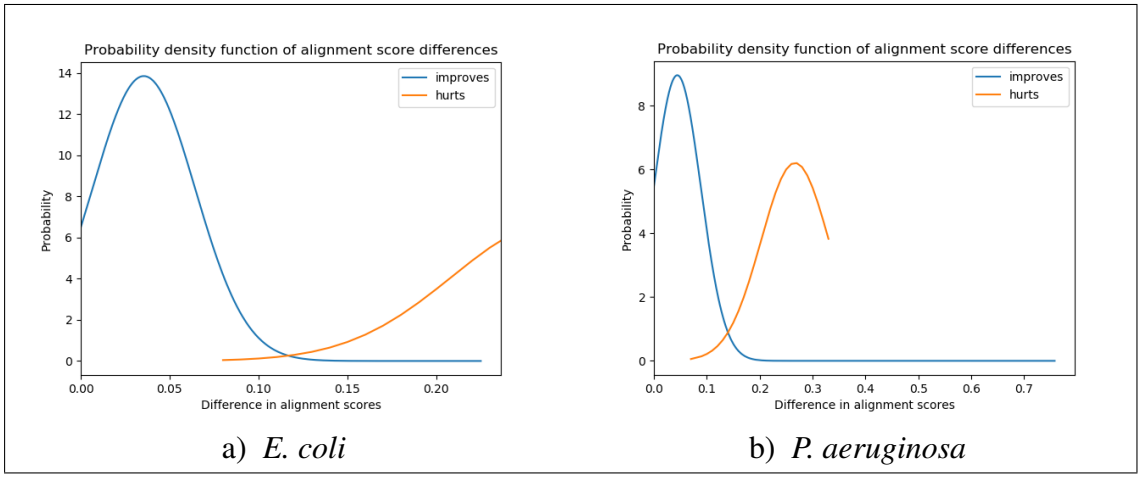
a) *S. pombe* b) *S. cerevisiae*

Figure 6: Probability density function of normalized alignment score differences between primary and secondary hits for the fungi from the ascomycota division with a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. The curves represent the density of the distribution of the data points at the indicated value on the x-axis, and the total area under each curve sums to one. Separation of the distributions in each figure demonstrates that including secondary hits when there is a lower normalized difference in alignment scores is likely to increase the number of true positives in results thereby also decreasing counts in the "false" categories.



a) *L. brevis* b) *C. difficile*

Figure 7: Probability density function of normalized alignment score differences between primary and secondary hits for the firmicutes with a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. The curves represent the density of the distribution of the data points at the indicated value on the x-axis, and the total area under each curve sums to one. Separation of the distributions in each figure demonstrates that including secondary hits when there is a lower normalized difference in alignment scores is likely to increase the number of true positives in results thereby also decreasing counts in the "false" categories.



a) *E. coli* b) *P. aeruginosa*

Figure 8: Probability density function of normalized alignment score differences between primary and secondary hits for the proteobacteria with a minimum alignment length filter of 200, a minimum mapping quality filter of 20, and a minimum category score filter of 40%. The curves represent the density of the distribution of the data points at the indicated value on the x-axis, and the total area under each curve sums to one. Separation of the distributions in each figure demonstrates that including secondary hits when there is a lower normalized difference in alignment scores is likely to increase the number of true positives in results thereby also decreasing counts in the "false" categories.

3.1.4 Alignment score ratios and chimeric hit processing

Similar tests using the ratio between alignment scores of primary and secondary hits was not a good indicator for predicting when secondary hits will help and hurt scores. Furthermore, considering only chimeric hits severely limited the data we were able to collect and provided little insight into when to include secondary hits. Therefore, while these parameters are still available in the final *PB&J* tool, we did not consider alignment ratios and chimeric hit processing when analyzing outputs from *PBtester*. We recommend keeping these parameters turned off until more phylogeny-based evidence is gathered, however in the future they may still prove to be useful depending on the species being studied and the goals of the researcher.

3.1.5 Open reading frame detection

As illustrated in Fig. 9, for low coverage data PALADIN's average execution times with ORF turned off are similar to those with ORF turned on. For high coverage data however, there was a huge increase in computation time when ORF was turned off. Since the focus of this research is low coverage data, this is not concerning. Fig. 10 demonstrates that there were not many outliers for the low coverage executions, and therefore we can be fairly confident in the averages plotted in Fig. 9. Thus we proved that we could get the highest quality results that PALADIN has to offer with minimal increase in computation time for low coverage data. Consequently, all PALADIN runs for the final data collected in this research were performed with ORF turned off, which is our final parameter recommendation for this pipeline.

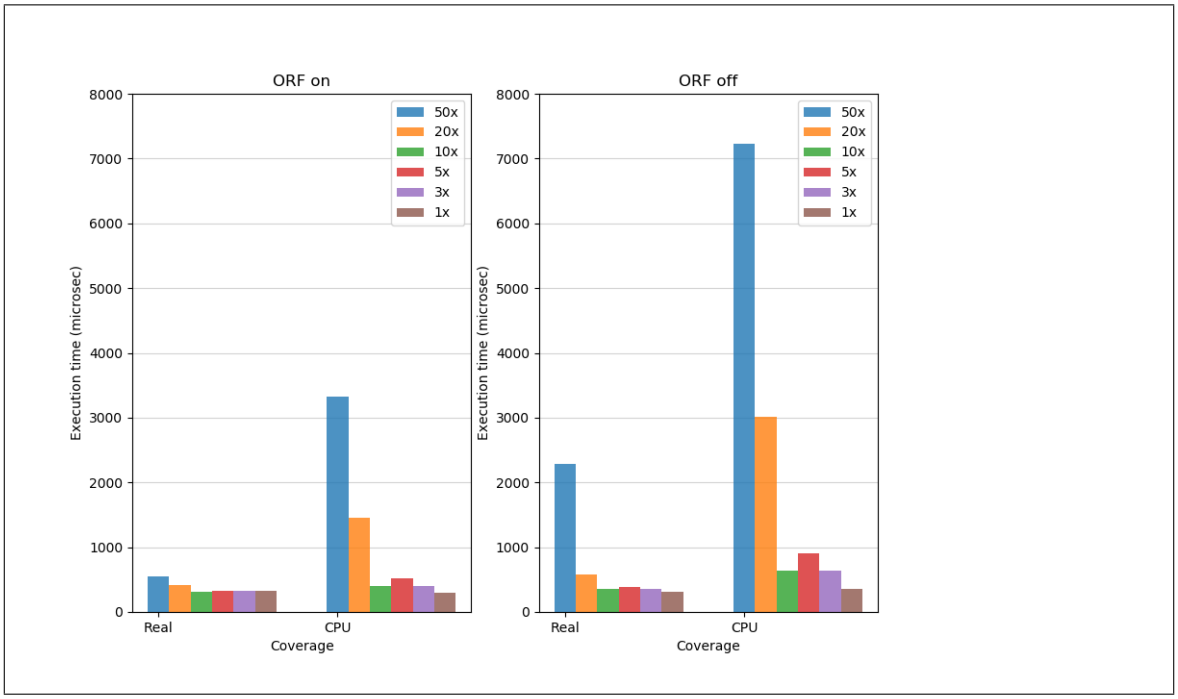


Figure 9: Average real and CPU times of PALADIN executions on ART-produced FastQ files for *C. gattii* at six different coverage levels. For each coverage level PALADIN was executed 100 times; 50 with ORF on, and 50 with ORF off, and the results were averaged.

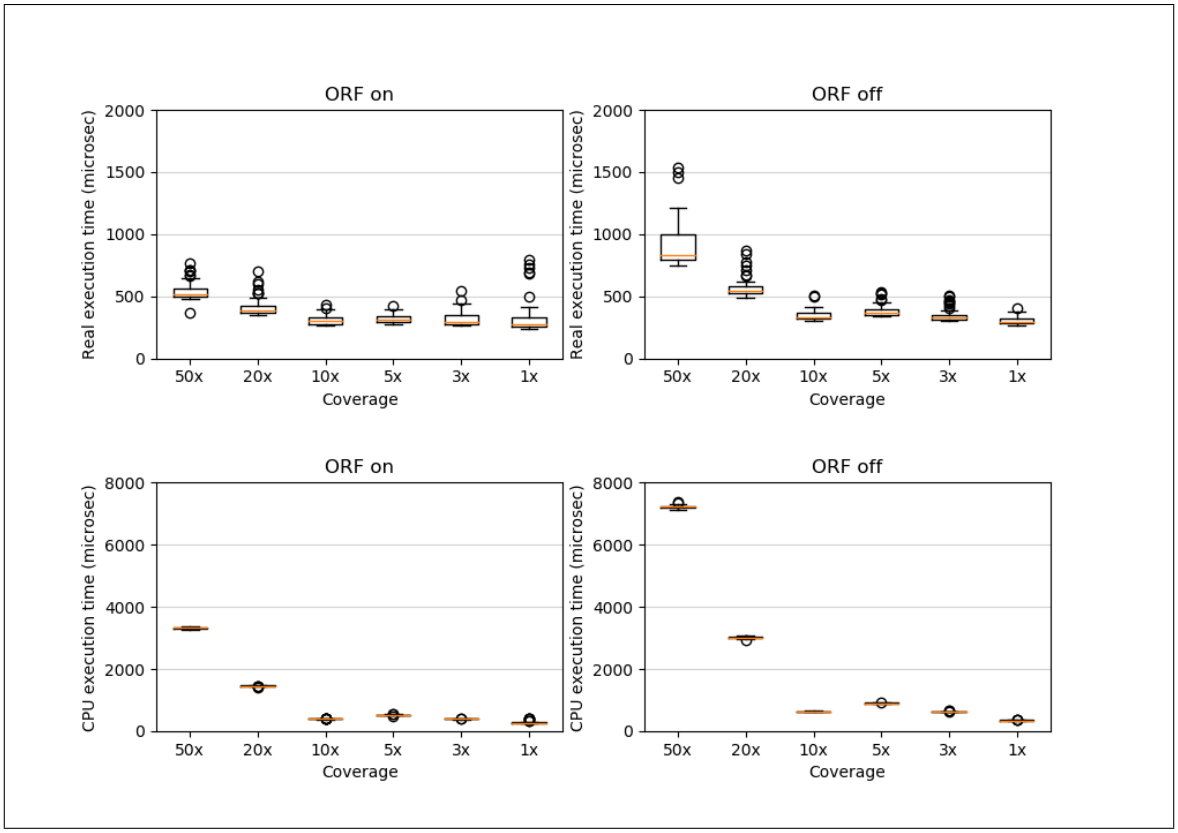


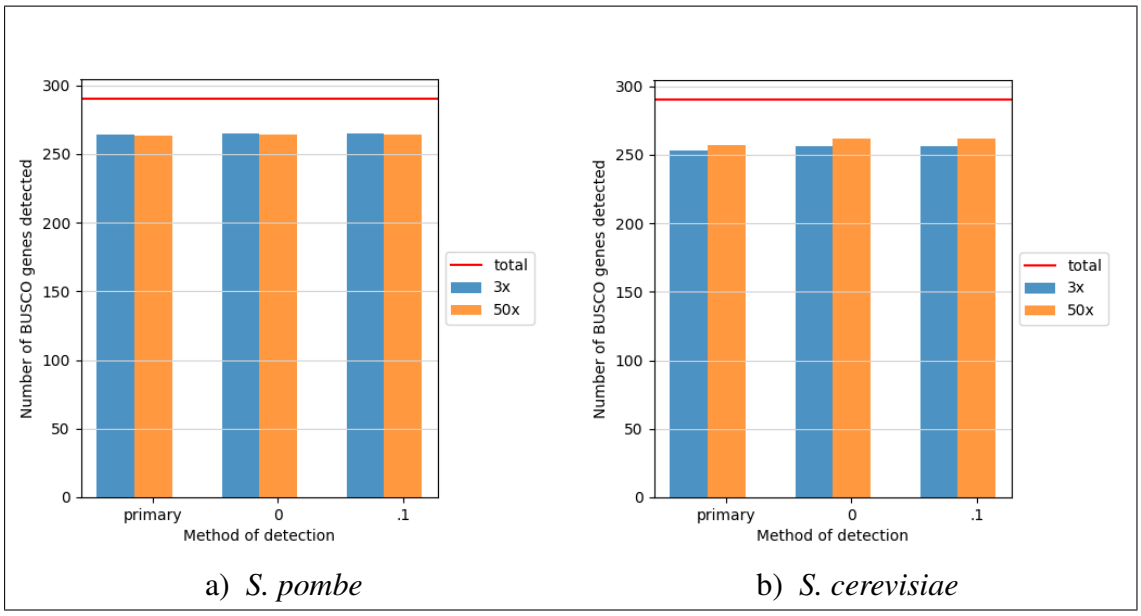
Figure 10: Visualization of outliers for the results shown in Fig. 9.

3.2 Assessment of genome completeness

Figs. 11, 12, and 13 demonstrate that 3x coverage data often results in similar genome completeness to 50x coverage data. Further, we may be able to make some tentative observations at the phylum level to provide phylogeny-based species and parameter recommendations to users.

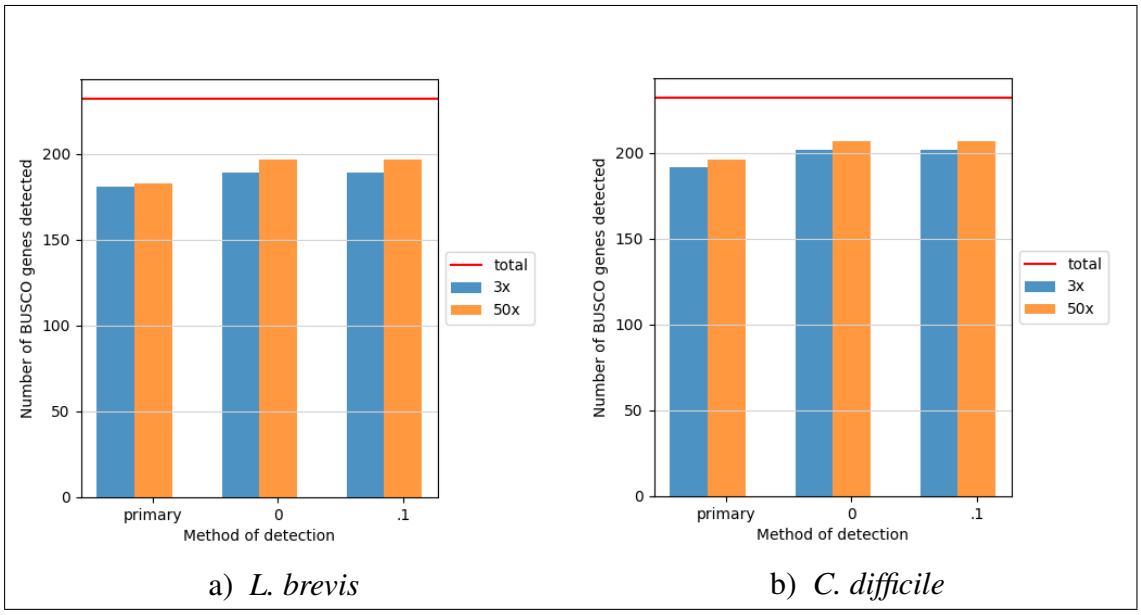
For the members of the ascomycota division that were tested, *S. pombe* and *S. cerevisiae*, PALADIN's performance was almost the same at 50x and 3x coverages (Fig. 11). Overall genome completeness of these two species as measured against the BUSCO benchmark was very high. Similar results are seen among the firmicutes we tested, *L. brevis* and *C. difficile* (Fig. 12). Adding in secondary hits for these two species improved their genome completeness approximately the same amount, and more than it improved that of the ascomycota. Together, these findings suggest that this tool may be useful for doing comparative genomics using low coverage data on species from the ascomycota division and the firmicutes phylum.

Members of the proteobacteria phylum, *E. coli* and *P. aeruginosa* were found to have lower genome completeness than the ascomycota and the firmicutes did (Fig. 13). This is especially the case when only primary hits were considered. Though adding in secondary hits for the proteobacteria increased their genome completeness considerably, their genome completeness at 3x coverage still remained low in comparison to their genome completeness when 50x coverage data was used. The fact that the proteobacteria have similar scores to one another overall suggests that this tool may not be the best choice for doing comparative genomics on proteobacteria using low coverage data.



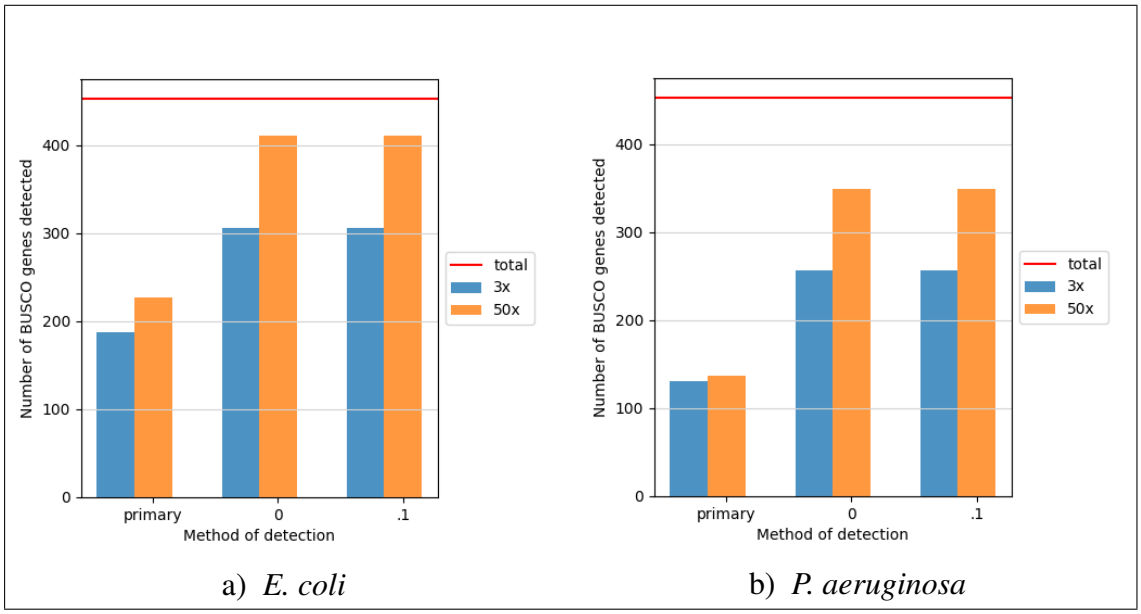
a) *S. pombe* b) *S. cerevisiae*

Figure 11: Genome completeness of the fungi species from the ascomycota division determined by using *PB&J*'s *score* command. A minimum alignment length filter of 200 and a minimum mapping quality filter of 20 were applied. Results collected when considering only primary hits are shown on the left. The two rightmost bar groups show PALADIN's results when considering secondary hits using minimum alignment score difference filters of 0 and 0.1. The red line represents the total number of BUSCO genes for the species as determined by BUSCO.



a) *L. brevis* b) *C. difficile*

Figure 12: Genome completeness of the firmicutes determined by using *PB&J*'s *score* command. A minimum alignment length filter of 200 and a minimum mapping quality filter of 20 were applied. Results collected when considering only primary hits are shown on the left. The two rightmost bar groups show PALADIN's results when considering secondary hits using minimum alignment score difference filters of 0 and 0.1. The red line represents the total number of BUSCO genes for the species as determined by BUSCO.



a) *E. coli* b) *P. aeruginosa*

Figure 13: Genome completeness of the proteobacteria determined by using *PB&J*'s *score* command. A minimum alignment length filter of 200 and a minimum mapping quality filter of 20 were applied. Results collected when considering only primary hits are shown on the left. The two rightmost bar groups show PALADIN's results when considering secondary hits using minimum alignment score difference filters of 0 and 0.1. The red line represents the total number of BUSCO genes for the species as determined by BUSCO.

3.3 Comparative matricies

After verifying the completeness of the genomes by using the *score* command, we used the *compare* command with a minimum quality filter of 20 to produce presence-absence matrices for groups of phylogenetically similar species. Percentages of genes shared across phylogenetically similar species are shown in Table 2. Some groups of species, for instance the firmicutes *L. brevis* and *C. difficile*, have a similar percentage of shared genes at 3x and 50x coverage. On the other hand, members of the proteobacteria phylum, *E. coli* and *P. aeruginosa* have twice the shared genes at 50x coverage than at 3x coverage as predicted by the lower completeness scores produced for them using the *score* command.

Species	3x	50x
S. pombe, S. cerevisiae	10.3%	17.1%
E. coli, P. aeruginosa	6.6%	10.8%
L. brevis, C. difficile	9.2%	11.2%

Table 2: Percentage of genes shared between phylogenetically similar species after applying a minimum quality filter of 20 to *PB&J*'s *compare* command.

CHAPTER 4

CONCLUSION AND FUTURE WORK

We demonstrated that for certain groups of species PALADIN alignments produced from 3x coverage reads exhibit comparable genome accuracy and completeness to those produced from 50x coverage reads and to the overall BUSCO benchmark. Thus for certain species PALADIN can be used with low coverage data to perform comparative genomic analysis. This approach is more cost-effective and less time consuming than traditional methods. Furthermore, we provide a simple tool, *PB&J*, and we recommend optimal parameters. This tool. when used in conjunction with PALADIN, simplifies the comparative genomic process.

In the next phase of research we plan to investigate more genome completeness trends at the phylum level to provide better phylogeny-based parameter recommendations. This will involve introducing more species into our pipeline, both from the same phyla as the species used in the current research, and from new phyla.

In a similar vein, we need to further quantify parameter recommendations. Since some users work with reads shorter than 250bp, this especially includes the minimum alignment length filter. Our preliminary tests with shorter minimum alignment lengths of 100 and 150 indicate that we can expect to get good reuslts with shorter reads, but this needs further verification.

We also need to do more low-level evaluation of the existing system. This includes looking at the specific genes in the cases where we fail, i.e., exploring gene sets in the "false" categories to see if they should be in a "true" category. It also means investigating and providing validation for the low comparison percentages in the compare phase by using external tools such as *OrthoFinder* [10] and *cd-hit* [11, 12]. We have already taken the first steps in this process, and

results so far seem to be confirming our existing findings. Nevertheless, more rigorous validation is still needed. Using the SAM file instead of the .tsv file from PALADIN may prove to be a better option for generating presence-absence data as it contains more information.

Finally, in the more distant future we may potentially investigate the extension of this tool to use in the metagenomics field.

CHAPTER 5

APPENDIX A

5.1 Genomes used

Several genomes from the original BUSCO paper [9] were selected for evaluation. Whole genomes from *Caenorhabditis elegans*, *Cryptococcus gattii*, *Schizosaccharomyces pombe*, *Saccharomyces cerevisiae*, *Escherichia coli*, and *Pseudomonas aeruginosa* were all run through the ART bioinformatics toolkit to generate single end Illumina reads at 1x, 3x, 5x, 10x, 20x and 50x average coverages.

To diversify the phylogeny of our results and enable us to identify patterns at the phylum level, ART runs were also performed on *Lactobacillus brevis*, *Micrococcus luteus*, *Microcystis aeruginosa*, *Clostridioides difficile*, and *Bacteroides fragilis*. 1x, 3x, 5x, 10x, 20x and 50x average coverages were again used.

CHAPTER 6

APPENDIX B

6.1 Programs

6.1.1 PB&J

The *map* command of this tool creates a BUSCO compatible data structure for PALADIN. Required parameters for this utility are an *ogs.id.info* file from the species' BUSCO database which contains the species' OrthoIds, and a *genes.tab* file, which contains the species' accession ids. Optionally, the user can specify the UniRef cluster level of their preference (UniRef 50, 90, 100, and unclustered are currently supported).

First, the program parses each line of the *ogs.id.info* file and populates a dictionary known as the "gene-ortho-dict". For every line, the gene id in column one is mapped to its OrthoId found in column two. Next the *genes.tab* file is parsed and a new dictionary known as the "gene-ortho-acc-dict" is populated. In this stage the gene-ortho-dict is queried for each gene id in the *genes.tab* file. Should the gene-ortho-dict contain a matching gene id, the gene id becomes a key in the gene-ortho-acc-dict. Its value consists of a tuple containing the previously-recorded OrthoId, and the accesssion id found in column four of the *genes.tab* file. Thirdly, for each gene id in the gene-ortho-acc-dict, the program uses PALADIN's *crosseref* plugin to look up the representative KBID for the current accession id. Each OrthoId is printed to the the OrthoKBID cross-reference file alongside the resulting representative KBID.

Score is the tool's second command, and uses the OrthoKBID cross-reference file produced by the *map* command to assess the number of BUSCO genes correctly identified by PALADIN. Required parameters are the paths to a PALADIN-produced SAM file and a BUSCO .tsv file, and the OrthoKBID cross-reference file.

The first step of this command's algorithm is to collapse data from the OrthoKBID cross-reference file back from UniprotIds to OrthoIds. Next, using this collapsed database, a set of unique OrthoIds is populated. Then, for each line of the SAM file, the program applies the user-specified filters, after which it gets the primary and secondary hit information of the current PALADIN alignment. A PaladinHit object is created from this information and stored in the main database for the program. If the user turned on secondary hit detection, a deep copy of the main database is made, and all primary and secondary true positives are aggregated.

In the next phase the BUSCO .tsv file is parsed, and from it a dictionary of sets is populated. The resulting BUSCO database contains separate sets of complete, fragmented, duplicate, and missing BUSCOs. Furthermore, sets of BUSCOs detected by PALADIN are populated from the main database and the deep copy of the main database which contains the aggregated primary and secondary true positives. The final step involves taking the intersection of each of the BUSCO sets in the BUSCO database with the sets of BUSCOs detected by PALADIN and reporting the results.

Compare is the third command provided by the tool, which creates a comparative matrix and list for any number of species specified by the user. The user must give the path(s) of one or more PALADIN .tsv files.

The algorithm for this command works as follows. All PALADIN .tsv files are parsed, and two separate dictionaries are maintained. The first, "genes-to-species", stores each unique gene id as a key, and a list of all species that had that gene id as the corresponding value. The second, "species-to-genes", does the opposite, recording the species as the key, and all genes that the species had as a list of values. After this preliminary stage, the program queries each key in the genes-to-species dictionary to determine all of the species that had a given gene. For each new query, a local set is populated with the appropriate species' columns to mark in the output file as having had the gene, and this information is written to the output file.

6.1.2 PBtracker

The purpose of *PBtracker* is to update the headers in a FastQ file to identify the exact source of each read produced by an ART run on a species. The program has four required arguments: the gff file for the species, the OrthoKBID cross-reference file, and the SAM and FastQ files from the ART run of the species being tested.

Step one of the algorithm used by this tool involves parsing the gff file and storing EMBL ids in a dictionary called the "CDS-info-map" along with their chromosome id and absolute start and stop positions, then using the *crossref* plug-in to map these EMBL ids to their representative KBIDs.

In the second phase, the representative KBIDs from this dictionary are compared to the ones in the OrthoKBID cross-reference file. Any ids that appear in both places are stored in a different dictionary known as the "CDS-info-KB-map".

Lastly, the FastQ and SAM files are parsed in conjunction with one another. If the chromosome id on a given line in the SAM file is found in the CDS-info-KB-map and the corresponding header overlaps with its previously stored start and stop position, the KBID mapped to this chromosome id, along with all of its information saved in the CDS-info-map, is written to the corresponding header in the FastQ file. We refer to this file as the modified FastQ file. Additionally, a list of the KBIDs that mapped to a chromosome id is written to a separate file known as the "*PBtracker* statistics" file.

6.1.3 PBtester

This tool parses the SAM file from a PALADIN run that used the modified FastQ file produced from *PBtracker* and produces a series of confusion matrices, overlap statistics, and statistics on secondary hits. Required arguments are the SAM file from the PALADIN run, the OrthoKBID

cross-reference file for the species in question, and the *PBtracker* statistics file. The algorithm used is a derivative of the one used by *PB&J*'s *score* command described in section 6.1.1.

Bibliography

[1] R. Ekblom, and J. Wolf., "A field guide to whole-genome sequencing, assembly and annotation," *Evolutionary Applications*, vol. 7, no. 9, pp. 1026–1042, Jun. 2014.

[2] Illumina. "Sequencing Coverage," *Coverage depth recommendations*, Illumina, Inc., 2019, https://www.illumina.com/science/education/sequencing-coverage.html.

[3] A. Westbrook, J. Ramsdell, T. Schuelke, L. Normington, R. D. Bergeron, W. K. Thomas, and M. D. MacManes., "PALADIN: protein alignment for functional profiling whole metagenome shotgun data," *Bioinformatics*, vol. 33, no. 10, pp. 1473-1478, May. 2017, 10.1093/bioinformatics/btx021.

[4] H. Li, and R. Durbin., "Fast and accurate short read alignment with Burrows-Wheeler transform." *Bioinformatics*, vol. 25, no. 14, pp. 1754-1760, May. 2009, 10.1093/bioinformatics/btp324.

[5] R. Waterhouse, M. Seppey, F. Simao, M. Manni, P. Ioannidis, G. Klioutchnikov, E. Kriventseva, and E. Zdobnov., "BUSCO applications from quality assessments to gene prediction and phylogenomics." *Molecular Biology and Evolution*, vol. 35, no. 3, pp. 543-548, Mar. 2018, 10.1093/molbev/msx319.

[6] W. Huang, L. Li, J. R. Myers. and F. T. Marth., "ART: a next-generation sequencing read simulator." *Bioinformatics*, vol. 28, no. 4, pp. 593-594, Dec. 2011, doi:10.1093/bioinformatics/btr708.

[7] SchedMD. "Slurm workload manager version 19.05," *Documentation*, 2019, https://slurm.schedmd.com/.

[8] S. Nurk, A. Bankevich, D. Antipov, A. Gurevich, A. Korobeynikov, A. Lapidus, A. Prjibelsky, A. Pyshkin, A. Sirotkin, Y. Sirotkin, R. Stepanauskas, J. McLean, R. Lasken, S. Clingenpeel, T. Woyke, G. Tesler, M. Alekseyev, and P. Pevzne. (April, 2013). Assembling Genomes and Mini-metagenomes from Highly Chimeric Reads. Presented at the Annual International Conference on Research in Computational Molecular Biology. [Online]. Available: https://link.springer.com/book/10.1007/978-3-642-37195-0

[9] F. Simao, R. Waterhouse, P. Ioannidis, E. Kriventseva, and E. Zdobnov., "BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs." *Bioinformatics*, vol. 31, no. 19, pp. 3210-3212, Jun. 2015, 10.1093/bioinformatics/btv351.

[10] D. Emms, and S. Kelly., "OrthoFinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy." *Genome Biology*, vol. 16, no. 157, Aug. 2015, 10.1186/s13059-015-0721-2.

[11] W. Li, and A. Godzik. "Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences." *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, Jul. 2006, 22:1658-9.

[12] L. Fu, B. Niu, Z. Zhu, S. Wu, and W. Li. "CD-HIT: accelerated for clustering the next-generation sequencing data." *Bioinformatics*, vol. 28, no. 23, pp. 3150–3152, Dec. 2012, https://doi.org/10.1093/bioinformatics/bts565.