RESEARCH AND DEVELOPMENT OF A WEIGHTED MOST RECENT COMMON ANCESTOR ALGORITHM FOR METAGENOMIC TAXONOMIC ASSIGNMENT

BY

HELLEN BUTUNGI

REG NO: I56/71340/2007

BSc. (MAK), MSc. (UoN)

A dissertation submitted to the Centre for Biotechnology and Bioinformatics (CEBIB) in partial fulfillment of the requirements for the Award of a Master of Science Degree in Bioinformatics of the University of Nairobi

University of Nairobi

2012

DECLARATION AND APPROVAL

I declare to the best of my knowledge that this dissertation is my original work and has not been presented for any degree or diploma in any other University.

HButungi

HELLEN BUTUNGI

This dissertation has been submitted for examination with our approval as University supervisors.

Dr. Gary Van Domselaar
Public Health Agency of Canada
University of Manitoba / University of Nairobi

Dr. Wanjiku Ng'ang'a
School of Computing and
Informatics
University of Nairobi

Dr. Etienne P. de Villiers
International Livestock
Research Institute (ILRI)

Nairobi

Prof. James O. Ochanda
Centre for Biotechnology and
Bioinformatics (CEBIB)

University of Nairobi

ABSTRACT

The new generation of metagenomics has gained tremendous popularity in recent years. This has been majorly due to rapid advances in DNA sequencing technology, which has produced large amounts of sequence data in relatively shorter times, compared to conventional DNA sequencing methods. There is a need to taxonomically characterise these data by assigning individual sequence reads to their constituent taxa. However, there is lack of up-to-date and customized software tools to accomplish this task, and for taxonomic characterisation, an automated taxonomic classification scheme is necessary. The overall objective of this study was to improve the accuracy of the most recent common ancestor (MRCA) estimation method used in scoring metagenomic reads in the pathogen profiling pipeline (PPP). The specific objectives included investigating sequence comparison algorithms that have been used for assigning sequence reads to taxa excluding the MRCA, compare the taxonomic classification accuracy of MEGAN and MRCA on the same simulated metagenomic dataset and finally design the weighted MRCA algorithm that attains the maximum possible classification accuracy and implement it in the PPP. A novel "weighted most recent common ancestor" (weighted MRCA) algorithm was developed as a number of Perl scripts and evaluated for taxonomic accuracy. The datasets used for evaluation were simulated by the QSA Read simulator using reference viral and prokaryotic (Bacteria and Archaea) genomes obtained from the NCBI Refseq database. The results showed an improved mapping of up to 3.6% for viral sequences and 8.4% for the prokaryotic sequences (*p*-values as low as 0.0043 at a significance level of $\alpha = 0.05$), at the species rank compared to MEGAN and MRCA. In the context of environmental science and medicine, these percentages are highly significant as they inform key decisions in public health. For large-scale pathogen discovery projects, this method facilitates more accurate analysis and reporting of candidate etiological agents in complex nucleic acid mixtures, which enhances outbreak preparedness by enhancing capacity for early recognition and containment of pathogens.

ACKNOWLEDGEMENTS

I am heavily indebted to the German Academic Exchange Service (DAAD) through the network of Natural Products for East and Central Africa (DAAD/NAPRECA) that provided me with the financial support, which enabled me undertake this degree. Additionally, I thank the World Federation of Scientists (WFS) through the National Scholarship Fund (NSF) for providing me subsistence funds while carrying out research.

To the then Vice-Chancellor of Makerere University, Prof Livingstone Luboobi, and the entire staff of Biochemistry Department, Faculty of Science; thank you for believing in me and granting me a study leave for without which I would not have been able to pursue these studies. The research skills and academic growth I attained during this time is immeasurable.

I am most grateful to Dr. Gary Van Domselaar, at the National Microbiology Laboratory (NML), Public Health Agency of Canada (PHAC) for giving me an opportunity to experience cutting edge research in his lab. I thank you for securing the funding for my research and for the helpful discussions that were committed to my progress. Your expertise as a research bioinformatician undeniably contributed to the successful completion of this project. Sincere gratitude goes to my other supervisors, Dr. Wanjiku Ng'ang'a, Dr. Etienne de Villiers and Prof. James Ochanda for allowing to guide me as I carried out this research. Thank you for dedicating your time to correcting this text and turning it into something beautiful. I am also indebted to Morag Graham and Shaun Tyler for helpful discussions and insights into metagenomics. Tom Matthews, words alone cannot say how much I appreciate all the tireless efforts that you put into the Pathogen Profiling Pipeline (PPP) project, which was the basis of my research. Even when I was in Winnipeg, you endured my questions every now and then, thank you so much. I can say that my Perl programming skills improved ten-fold. The skills you imparted in me hugely made this study a success. Special thanks go to the entire Bioinformatics core at the NML; Phil for the Perl tutorials, Heather, Shane, Matt, Franklin, Ben and my fellow CO-OP's Aaron, Abid, Josh, Wayne, Eric. You guys are awesome.

To Dr. Songok and family, thanks for the hospitality. To my beloved host Joyleen, thank you for having me. I cannot even bring myself to think of what my stay in Winnipeg could have been without you. The concerts, walks, dinners were all refreshing to my memory, and I needed them so badly. You were my councillor, my sister and my best friend. Thank you so much Carol for all the laughs we had, those moments will always be in my memory. May God reward you abundantly.

My friend Wanjugu, you are irreplaceable in my heart. Thanks for showing me the "Kenya". To Alan Orth at ILRI, *unanibamba.* The Linux kung fu I learnt from you will always remind me of what a great person you are and of your willingness to share with others. *Asante sana.*

I am extremely thankful for my parents, Rev and Mrs. Baraire for your endless prayers and encouragement throughout the Master's program. Thank you for believing in me and always pushing me to be better. You are the world's best parents and I love you so much. To my friends, sisters, thank you for your love and prayers.

Most importantly, I thank the Almighty God for granting me good health, and strengthening me throughout this study. For in His word He says, *"For I know the plans I have for you," declares the LORD, "plans to prosper you and not to harm you, plans to give you hope and a future"* (Jeremiah 29:11) .

DEDICATION

To my beloved parents, Rev. Stanley and Mrs. Sarah Baraire.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BLAST	Basic Local Alignment Search Tool
BLASTN	Nucleotide BLAST
CPAN	Comprehensive Perl Archive Network
rDNA	Ribosomal DNA
EGT	Environmental Gene Tag
FASTA	Fast Alignment
FN	False Negatives
FP	False Positives
FTP	File Transfer Protocol
GC	Guanine-Cytosine
GOS	Global Ocean Sampling
GS FLX	Genome Sequencer FLX system
HSP	High-scoring Segment Pair
HSOM	Hyperbolic Self Organizing Map
IMM	Interpolated Markov Model
MEGAN	MEtaGenome ANalyzer
MRCA	Most Recent Common Ancestor
NCBI	National Centre for Biotechnology Information
NML	National Microbiology Laboratory
Next-gen	Next generation
OTU	Operational Taxonomic Unit
Pfam	Protein families
PHAC	Public Health Agency of Canada
PID	Percent Identity
PPP	Pathogen Profiling Pipeline

rRNA	Ribosomal RNA
SOrt-ITEMS	Sequence ORTholog based approach for binning and Improved Taxonomic Estimation of Metagenomic Sequences
SOM	Self-Organizing Map
S-GSOM	Seeded Growing Self Organizing Map
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives
2D	Two Dimensional

CHAPTER ONE: INTRODUCTION

1.1 General Introduction

The new generation of metagenomics has brought with it a wealth of knowledge and changed the perspectives of analyzing microbial genomes. Metagenomics has been defined as "the study of the Deoxyribonucleic Acid (DNA) of uncultured microorganisms" (Handelsman, 2004). A *genome* is the entire genetic information of one organism, whereas a *metagenome* is the entire genetic information of an assemblage of organisms (Handelsman, 2004). This new approach utilizes techniques and protocols that allow researchers to obtain, and sequence the genomic content of microbial communities directly, without the need for culturing and cloning of all individual organisms present in the sample (Hugenholtz, 2002). Computational methods are subsequently used to study the sequenced fragments by numerous characterization strategies. (Wooley *et al.*, 2010). In some cases, sequence fragments may be assembled into contigs that can be aligned using software and studied as whole genomes (Raes *et al.*, 2007). Bypassing the need for isolation and laboratory cultivation of individual species, we can in principle be able to access 100% of the genetic information available in a microbial community in contrast to merely 1% accessible through traditional sequencing of culturable organisms (Ghazanfar *et al.*, 2010). . This new and evolving field of biology has demonstrated a wealth of comprehensive power and richness and is making tremendous contributions to microbial ecology, biodiversity, bioremediation, bioprospection of natural products, medicine, and many other fields (Ghazanfar *et al.*, 2010).

The main goals of a metagenomics analysis include identification of what populations of microorganisms are present in a given sample ("who is out there?") and determining the role that each microorganism plays within a specific environment ("what are they doing?") (Handelsman, 2004). Metagenomics samples are found almost everywhere in the environment within the ocean, soil samples and several places within the human body (Turnbaugh *et al.*, 2007), the diversity of microorganisms is thus thought to be in the range of tens of millions to greater than hundreds of millions of species (Harkins *et al.*, 2007). This presents a bottleneck in the analysis due to the complexity and diversity of the genomes involved as opposed to single-species genome analysis. Each of the genomes in the metagenomic sample has to be associated with its source organism, a task that is daunting. Since some microbial genomes have previously been sequenced, by extending the sequence data present in public repositories, it is possible to get a glimpse of what microorganisms are present in the sample. Some of the sequenced fragments may correspond to new organisms, genes and proteins that have potential applications in biotechnology and medicine (Steele and Streit, 2005).

Metagenomics and counterpart meta-strategies have achieved tremendous popularity because of two major developments. There has been extensive (still ongoing) establishment of high-throughput DNA sequencing centres, employing next generation (next-gen) technologies (Petrosino *et al.,* 2009). This has enabled the sequencing of large datasets in a relatively short time, an idea that had been difficult to realize. Moreover these datasets can be obtained at a relatively low cost compared to earlier sequencing technologies (Petrosino *et al.*, 2009).

1.2 Strategies in metagenomics sequencing

Two metagenomic sequencing strategies, all aimed at identifying microbes in a complex community can be distinguished: directed metagenomics and shotgun metagenomics.

1.2.1. Directed metagenomics

Directed metagenomics involves the sequencing of long insert libraries after screening for the presence of certain phylogenetic e.g. 16S ribosomal Ribonucleic Acid genes (16S rRNA genes) or functional (e.g. certain enzymatic activity) markers (Sogin *et al.,* 2006). There exists gene sequences in the small subunit of ribosomal RNA (16S rRNA genes) that play a significant role in estimating phylogenetic diversity and taxonomic mapping of environmental populations (Sogin *et al.,* 2006). This is because there occurs significant variations in these rRNA genes of different organisms (phylotypes) that allow for their differentiation. The highly conserved nature of these regions of sequence implies that Polymerase Chain Reaction (PCR) amplification with universal primers can allow for annealing of only these regions of sequence to the primers (Pace, 1997). Subsequent cloning of these individual genes allows for separation, in which case the genes can then be sequenced. A filtering step to remove primers and other low quality data leaves only unique RNA tag sequences, which when analyzed generate a wealth of information relating to species richness and the relative abundance of the different Operational Taxonomic

Units (OTUs) present in the microbial sample. Each of the tag sequences corresponds to an individual OTU (Pace, 1997). A multiple alignment of tag sequences against the RNA sequences in the Ribosomal Database Project II (Cole *et al.*, 2007) followed by subsequent clustering using linkage algorithms (Schloss and Handelsman, 2005) can generate a phylogenetic tree that clearly relates all the OTUs in the sample. The amplification of 16S rRNA genes in metagenomics studies is prone to sequencing errors and other errors resulting from formation of chimeric sequences (Ashelford *et al.*, 2005). These have been found to contribute largely to anomalies in public rRNA sequence databases. In one of the studies by Quince and colleagues (Quince *et al.*, 2009), there was evidence of overestimation of species diversity due to these errors. In order to eliminate these sequencing errors and maximize accuracy, Petrosino *et al.*, (2009) propose the use of proofreading DNA polymerases, and the variation of temperature gradients during the PCR amplification stage. This has resulted in a marked improvement and specificity of the DNA amplification, which in turn lessens the effect of subsequent errors in the sequencing stage that follows thereafter.

1.2.2 Random shotgun sequencing

16S rRNA sequencing which has for many years been considered as the yardstick for efficient characterization of microbial communities may not serve to discriminate against and subsequently detect rare members of a given microbiome (Petrosino *et al.*, 2009). This stems from the fact that microbiomes tend to be very divergent in terms of microorganisms present (Petrosino *et al.*, 2009). Thus, in order to overcome these challenges, whole genomes providing more comprehensive genome coverage can be sequenced. This is done by the use of the new generation parallel high-throughput sequencing technologies. The nature of these shotgun sequencing approaches has led to the discovery of many new genes that encode different biochemical and metabolic functions, as evidenced in the "Global Ocean Sequencing project", (Yooseph *et al.*, 2007). In this strategy, random sequencing of clones generated from aggregate DNA by Sanger or pyrosequencing of aggregate DNA, without cloning is carried out. DNA is first randomly sheared into smaller fragments which are then sequenced individually to obtain "reads." This unbiased approach provides a broad survey of almost all the gene content and metabolic capabilities of a given microbiome. A major challenge in shotgun metagenomic sequencing projects is the potential contamination of metagenomic samples with host genetic material. Due to the sources from which these samples are drawn, and the manner in which they are treated (extract and shred all DNA in the sample), it is inevitable to avoid the effects of host microbial contamination (Kunin *et al.*, 2008). However, different subtraction strategies (for human DNA sequences) are being developed to lessen the effect of host DNA contamination (Petrosino *et al.*, 2009). More recently, a standalone method as well as its counterpart web-based version has been developed to identify and subsequently remove contaminants from sequence data (Schmieder and Edwards, 2011). Also, since most of the DNA corresponds to potentially previously uncharacterized genes, shotgun metagenomic projects result in a high number of genes of unknown function. This is mainly due to the fact that similarity searches against current public repositories may not always contain relatives to the target sequences being searched for (database limitation) as highlighted in Huson *et al.*, (2009). Evidence from recent studies (Shah *et al.*, 2010; Morgan *et al.*, 2010) has shown that taxonomic profiles inferred from metagenomic sequences greatly rely on and are limited by the DNA extraction method used as well as the sequencing protocol employed. Furthermore, shotgun metagenomic sequencing is generally not deep enough to detect rare species in complex communities but can be extended more reliably to identify viruses, bacteria, fungi and protists. Notable projects based on these methodologies include data sets from an acid mine biofilm (Tyson *et al.*, 2004), seawater samples (Venter *et al.*, 2004; DeLong *et al.*, 2006), deep-sea sediments (Hallam *et al.*, 2004), or soil and whale falls (Tringe *et al.*, 2005).

Random shotgun metagenomics has enabled the rapid characterization of novel and emerging pathogens at the genomic level, providing new means for unbiased and unambiguous identification of microbial disease agents in situations where other diagnostic technologies may fail. Moreover, studies are now emerging that demonstrate the power of next generation (next-gen) sequencing technology in identifying previously elusive etiological agents. For example, this technology has been applied in outbreak surveillance to identify novel viruses associated with Colony Collapse Disorder in honeybee colonies (Cox-Foster *et al.*, 2007), and also used as a diagnostic tool for detecting etiological agents in transplant-associated fatalities (Palacios *et al.*, 2008).

1.3 Problem statement

Taxonomic characterization of metagenomic data involves assigning individual sequence reads to their source organisms, which requires customised software tools. The PPP utilises the MRCA algorithm to perform this task. The method is simple, fast, and reasonably accurate. However, it fails to take into account possible confounding factors such as sequence complexity, horizontal transfer of sequences across species, and the degree of representation of identical sequences within other microbes. It also fails to consider taxonomic assignments within the context of the larger overall read data set, wherein assignments may be supported by other (very numerous) read taxonomic assignments.

1.4 Justification

A typical metagenomics study produces a massive amount of data that comprises a diverse population of microorganisms. The exploratory nature of this approach and the lack of customised software solutions to analyse this data presents a substantial challenge for bioinformatics analysis towards obtaining meaningful information from these large, complex datasets. Many studies still make use of classical methods, outdated software or web services that originally were not intended for metagenomic data analysis and thus have to be adapted or pipelined to produce the desired results (Pachter, 2007). There is thus a need to develop up-to-date customised computing and targeted analysis systems that quickly organize metagenomics sequence output data for quick and logical characterization of these microbiomes. Owing to the massive amounts of data generated, an automated taxonomic classification scheme is necessary, and because it is automated, it must have the maximum possible classification accuracy.

The weighted MRCA algorithm is implemented in the Pathogen Profiling Pipeline (PPP) which is an important bioinformatics tool for metagenomics analysis. Designed for application in the earliest stages of disease outbreaks and in situations where standard diagnostic technologies may fail to conclusively identify a causative agent of infectious disease, PPP provides a comprehensive and flexible system for biologists to rapidly identify candidate etiological agents, including novel unknowns, directly from clinical specimens for follow-up confirmatory studies.

1.5 Objectives

The main objective was to improve the accuracy of the MRCA estimation method used in scoring metagenomics reads in the PPP.

The specific objectives were to:

i. Investigate sequence comparison algorithms for metagenomic taxonomic assignment, excluding MRCA.

ii. Compare the taxonomic classification accuracy of MEGAN and MRCA on a simulated metagenomic dataset generated using QSA Read Simulator.

iii. Design the weighted MRCA algorithm that attains the maximum possible classification accuracy and implement it in the PPP.

1.6 Motivation

This work was done within the context of the metagenomics software research and development program headed by Dr. Gary Van Domselaar at the National Microbiology Laboratory (NML) of the Public Health Agency of Canada (PHAC), in Winnipeg Manitoba. Work on the development of computational tools for metagenomics analysis was initiated as a number of ad hoc Perl scripts to assist in analysis of the enormous in-house generated metagenomic data from pyrosequencing projects at PHAC. PPP was developed and has since evolved into a fully fledged system with a customisable user interface that can be used in metagenomic analysis from sequence analysis to generating informative reports. The ability to correctly identify the source organism from which a given genome sequence read derived from a myriad of genomic data (metagenome) is the utmost goal. Initial work on the PPP revealed that the MRCA algorithm used for scoring reads had several limitations, the most severe of which was the inability to exclude outlying (i.e. taxonomically distant) candidate taxa from consideration when assigning a group of similar reads to an OTU.

1.7 Scope of the study

This work focused on extensively reviewing other methods (algorithms) that have been used for taxonomic estimation of metagenomic samples and deliberating on which aspects of these methods can be exploited for implementation within the PPP to augment or replace the existing algorithm. Both sequence comparison and sequence composition methods for metagenomic taxonomic assignment were assessed in that order. The methods that could be utilized for taxonomic estimation of viral and prokaryotic (Bacteria and Archaea) sequence data were tested for taxonomic classification accuracy. The QSA Read simulator was used to generate short sequence reads of lengths 200-500bp (simulated test data sets), in a manner that is in accordance with pyrosequencing data as obtained from the GS FLX genome sequencer. Test metagenomic datasets were generated from reference genomes of all viral and prokaryotic (bacteria and archaea) origin obtained from the NCBI Refseq database (release 41). All tests were run on the same Intel(R) Xeon(R) CPU, 1.86 GHz processor workstation running Ubuntu 10.04 x86_64. On the basis of classification accuracy, none of the reviewed methods surpassed the MRCA; a novel algorithm was designed, implemented in the PPP and assessed for taxonomic classification accuracy in a similar manner.

1.8 Thesis overview

Chapter one gives a general introduction into the field of metagenomics. Different strategies of metagenomic sequencing are briefly discussed here. The problem, justification and objectives of the study are also covered in this chapter. Chapter two focuses on reviewing other metagenomic analysis approaches for taxonomic estimation that have been used previously. This section highlights the details of the PPP software system which is used for testing in this project and its current limitations. The methodology used for this study is detailed in Chapter three with other installation guidelines and scripts in the Appendices. Results of the study and the discussion are combined in Chapter four. Conclusions, limitations and further work feature in Chapter five. Bibliography and Appendices for the study are reflected in Chapters six and seven respectively.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

Traditional genomic studies involve culturing, cloning and subsequent characterisation of individual microorganisms in the laboratory (Amann *et al.*, 1995). In the laboratory, the process of detecting pathogenic microbes in a clinical sample involves routine isolation, growing on specific biochemical media followed by subsequent testing using standard laboratory protocols. However, the biggest percentage of microorganisms found in natural environments cannot be identified by these methods (Petrosino *et al.*, 2009). This is because these microorganisms have not been previously cultured, thus the conditions that favour their growth on media are unknown. Staley and Konopka (1985) carried out experiments which led to the conclusion that greater than 99% of most microbes cannot be easily cultured and identified by these standard techniques. These unculturable individuals constitute a diverse population of microorganisms with distinct inter-relationships in the ecosystem, and are either not or distantly related to the culturable ones. Following from these findings, culture-independent methods (Reisenfeld *et al.*, 2004) are now preferred techniques for understanding genetic diversity and taxonomic relationships between these microorganisms within the ecosystem. A large number of microbes have previously been isolated, cultured and sequenced, resulting in an enormous amount of genomic sequence data that is deposited in public databases (Pruitt *et al.*, 2005; Sayers *et al.*, 2009; Benson *et al.*, 2011). This has strengthened the fields of microbiology and also impacted largely on microbial evolution studies. Scientists are now able to easily examine both the inter- and intra-relationships of these microorganisms, and draw insights into the functionality of the ecosystems from which these microbes are derived (Wooley *et al.*, 2010).

Manichanh *et al.*, (2008) demonstrated that close scrutiny and evaluation of a metagenomic dataset derived either by directed sequencing or shotgun approaches provides insights into the underlying genetic and microbial diversity stored in a metagenomic library. Venter and colleagues (2004) carried out a pilot study on whole genome shotgun sequencing to samples of the Sargasso Sea in order to characterize the microbial community and identify new genes and species. The dataset revealed extraordinary biodiversity including 1.66 million sequences comprising 1.045 billion base pairs (bp). The samples contained approximately 1800 microbial species, 150 new bacterial species and about 1.2 million new genes (Rolf and Carola, 2009).

The introduction of parallel, high-throughput sequencing technologies, resulting in the generation of huge amounts of data (millions of bases per run), has contributed to their widespread adoption. The rate at which these data are produced has also contributed largely to their popular usage backed by extensive commercialization. When compared to previous sequencing technologies such as shotgun or Sanger sequencing (Sanger *et al.*, 1977), they have much more popular usage (Petrosino *et al.*, 2009). The next-gen technologies that have been used widely to generate and sequence reads are the Roche/454 (Margulies *et al.*, 2007), Illumina/Solexa (Cuddapah *et al.*, 2009), Life/APG (Valouev *et al.*, 2008) and Helicos Biosciences (Harris *et al.*, 2008). The first generation instrument from Roche/454, the Genome Sequencer 20 (GS 20) generated 100-bp reads and 30 – 60 Mega bases (Mb) per run. The second (GS FLX Standard) and third (GS FLX Titanium generation platforms, yield 250-bp reads (approximately 150 Mb/run) and greater than 350 bp reads (approximately 400 Mb/run) respectively (Petrosino *et al.*, 2009).

Of interest is the 454 sequencing technology that was employed in generating sequence reads that were used in background studies for this work. The term *454 sequencing* refers to high-throughput sequencing platforms (e.g., Roche/454 Life Sciences) for metagenomics that are based on pyrosequencing chemistry. The pyrosequencing methodology used by the GS FLX instrument which was employed in this study, is based on protocols developed by Margulies *et al.*, (2007). Unique to this method is the usage of emulsion based PCR (emPCR) protocols in which each bead binds a single fragment of DNA (Margulies *et al.*, 2007). Since approximately one million of these beads can be deposited on a microtitre plate, a large number of fragmented DNA can be analyzed in a rather short time.

The sequencing step generates reads of approximately 400-500 nucleotides in length that have been randomly sampled from the genetic material contained in the metagenomic sample. In order to maximize the sampling of all taxonomies represented in the sample many sequence reads (up to a million) are generated. Overlapping sequence reads obtained may be progressively assembled into contigs by several assembly methods. The GS FLX instrument comes with an assembly program, GSAssembler read assembly. Alternatively, sequence reads can be analyzed directly and the taxonomic composition determined. Several methods have been developed for this purpose and differ in their taxonomic assignment depending on the algorithms employed and how the input DNA is represented (either as raw sequence reads, contigs or other segments such as a short sequence of nucleotides (oligomers)). These methods are reviewed in the next section.

The correct assignment of every read to its source organism is an important task in metagenomics and has been the subject of research in recent years. This process termed taxonomic profiling (Monzoorul *et al.*, 2009) involves the use of sequence similarity or sequence composition based methods. In their most popular usage, sequence similarity based methods use sequence similarity programs that relate the reads to sequence data represented in public databases. If the environmental DNA sequenced is closely related to sequence data from previously sequenced organisms, of which their sequence data is represented in the database, then the taxonomic profile of this metagenome can be easily and reliably estimated. However, as is usually the case with most metagenomic studies, most of the microbes in the environmental sample correspond to new organisms, due to constant evolution. Some of these microbes might have divergently evolved that no sequences in the public repositories closely relate to them. In such a case, a metagenomic study will result in most of these microbes being characterized as unknown (Huson *et al.*, 2009).

The process of obtaining OTUs, which are the taxonomic groupings present in the sample, from raw sequence reads, involves first surveying the reads obtained from pyrosequencing for possible contamination. In metagenomics, nucleic acid contamination may result from host organism or from the environment itself. This may be done by comparing the reads against a database containing host genome sequences. DeconSeq, a publicly available software to assess and subsequently remove contaminants from read sequences has recently been developed (Schmieder and Edwards, 2011). Sequence reads can also be filtered to remove duplicates and low quality reads. Sequence similarity programs e.g BLAST are then used to obtain regions of similarity between reads and reference database sequences e.g NCBI-nr, nt or RefSeq by mapping the read sequences against these databases. This yields a count of sequence reads assigned to a given organism. BLAST results are then evaluated in terms of parameters including percent identity, High-scoring Segment Pairs (HSPs), or bitscore values. Depending on the set threshold values for these parameters, the reads that fall within the range are preserved as high scoring sequence matches. The organisms to which the query reads mapped are then assessed for their taxonomic relationship by first mapping them to their corresponding taxonomic units in the NCBI taxonomy tree. Consequently, all the reads assigned to a particular source organism are grouped together. All the organisms that map to the same subtree in the taxonomy tree are grouped in the same OTU. Several approaches have been cited that give an estimate of taxa present in a metagenomic sample and they are broadly classified into two.

i. Sequence composition based and

ii. Sequence comparison based methods.

2.2 Sequence composition-based methods for taxonomic assignment

These methods extract sequence features, like GC content or k-mer frequencies, and compare them with features of reference sequences with known taxonomic classifications (Woese *et al.*, 1977). More particularly, different techniques, like the calculation of correlation coefficients between oligonucleotide patterns (Tatusov *et al.*, 2001), self-organizing maps (SOMs) (Cicarelli *et al.*, 2006), or support vector machines (SVMs) (Cole *et al.*, 2005) can be used to classify the metagenomic fragments. PhyloPythia is an example of a multiclass support vector machine (SVM) classifier that is reported to have achieved a classification accuracy of between 79–96% for fragments of unknown organisms (McHardy *et al.*, 2007).

In previous studies (Deschavanne *et al.*, 1999), genome sequence composition was used to phylogenetically characterise sequence fragments of unknown taxonomic origin. Sections of the genome sequence carry genome signatures that have been revealed to play a significant role in determining organism-specific characteristics. It is this characteristic feature that was utilized in the design of the PhyloPythia algorithm, alongside the oligonucleotide composition of variable-length genome fragments. These genome signatures are used to train an SVM which can then be applied to another dataset that may contain similar features. This high-dimensional supervised classification method uses sparse input data to solve the problem of phylogenetic assignment to known clades (dominant sample populations or higher level clades). PhyloPythia was tested on two fundamental metagenomic datasets; the Enhanced Biological Phosphorus-Removing (EBPR) sludge (Martin *et al.*, 2006) and the Sargasso Sea (Venter *et al*., 2004). In both cases, the method accurately classified genomic fragments ≥1–3 kilo bases (kb) for all taxonomic ranks considered. For fragments of unknown organisms, PhyloPythia was found to correctly assign greater than 80% for all lengths and taxonomic ranks for the query datasets. It was also noted that accuracy increased further when assigning fragments from known organisms. This is due to the fact that close relatives of these organisms exist in the databases used for training. For fragments ≥3 kb, the sensitivity and specificity was greater than 90% for clades from the rank of domain to order.

CompostBin (Chatterji *et al.,* 2008), a binning algorithm was developed for the purpose of solving the taxonomic classification problem for metagenomic samples of unknown origin. This unsupervised approach does not require training on currently available genomes and thus eliminates the database limitation where in environmental sequences can only be classified based on what is represented in the database. It uses a weighted version of the standard Principal Component Analysis (PCA) technique (Jolliffe, 2002) to extract a "meaningful" lower dimensional sub-space. Chatterji *et al*., (2008) report that raw environmental sequence reads and information about mate-pairs obtained from pyrosequencing are required for taxonomic assignment. Phylogenetic markers are also an important aspect and in combination with the above, provide the input sample to the CompostBin algorithm. The algorithm also utilizes some information about the possible number of abundant species, to be able to determine the number of bins in the output. Sample reads can be evaluated for rRNA marker genes as in all directed metagenomics projects to ascertain taxonomic groupings (Rusch *et al*., 2007). The algorithm can distinguish sequences from various species using just the first three principal components, aided by the normalized cut clustering algorithm (Chatterji *et al.,* 2008). CompostBin was applied to a simulated metagenomic dataset and resulted in accurate and definite taxonomic bins, even when applied to raw reads of short sequences. The error rates observed correlated mostly with the phylogenetic distances between the species and the relative abundance of species.

TETRA implements an unsupervised approach by utilizing the innate but weak phylogenetic signal carried in tetranucleotide frequencies. Reverse complements of both the forward and backward strands are obtained in either direction, to account for different codon usage. A maximal-order Markov model is used to calculate the frequencies of all 256 possible tetranucleotides and the corresponding expected frequencies from the sequences' di- and trinucleotide composition (Teeling *et al.*, 2004). This also helps detect tetranucleotide over- and underrepresentation by approximation (Schbath *et al.*, 1995; 1997) which converts the divergence between the observed and expected tetranucleotide frequencies into z-scores. As a last ditch, all DNA sequences are compared in pairs by computing the Pearson's correlation coefficient of their z-scores. TETRA performs comparably similar to PhyloPythia in identifying most of the larger fragments of the dominant sample populations. However, PhyloPythia surpasses in its ability to assign short fragments of the dominant populations and fragments of the higher-level clades that are best described by more complex shapes in feature space (McHardy *et al.*, 2007).

In studies by Abe *et al.*, (2002, 2005, 2006, 2007), the standard Self Organizing Map (SOM) showed the ability to classify environmental DNA fragments of lengths >10 kb. This was done by projecting the data into a two-dimensional (2D) flat Euclidean space, which enables visualising several features at once. On the map, every lattice point represents a node whose weight has the same dimension as the input vector (oligonucleotide pattern). Similar samples are clustered together on the grid map and unknown sequence data can be classified by calculating the distance between the lattice points (Abe *et al.*, 2007). However, genomic intrinsic features may not be mapped correctly in this space and it is more likely that they are structured hierarchically. This is the same representation of species in the Tree of Life. The importance of organizing data in a hierarchical manner cannot be surpassed as it enables the data to grow exponentially (Hierarchical SOM). In a similar manner, the amount of metagenomic data grows exponentially, and this implies that mapping into a geometric space with similar behaviour is more appropriate. This method has been previously applied to text mining by Ontrup and Ritter (2006), and registered many successes. The same strategy continues to show promise for genomic sequences, as previously documented by Martin *et al.*, (2008).

The seeded-Growing Self Organizing Map (S-GSOM) has also been used to analyze metagenomic data and can form bins at different taxonomic levels. As opposed to SVMs, the S-GSOM is a semi-supervised clustering algorithm, utilising similar features in the sequence data to form taxon-specific bins. Like all binning methods, bins that have different labels in a lower taxonomic level may belong to the same higher taxonomic level but can be combined to form higher taxonomic bins, thus accuracy is higher, at the cost of lower taxonomic resolution (Chan *et al.,* 2008). S-GSOM outperformed the binning methods that depend on already-sequenced genomes, and compares well to the current most advanced composition-based binning method, PhyloPythia.

Supervised algorithms e.g SVMs, perform better at a given classification problem since they have been trained on the inherent distinguishing characteristics of the dataset as opposed to the counterpart unsupervised methods that possess no knowledge of relevance of features (McHardy *et al.,* 2007). However, some of the tools above require training, employing known genomic sequences of different taxonomic origin. The accuracy of the phylogenetic classification thus depends on a number of factors such as fragment length of the environmental DNA and the amount or origin of the genomic sequences used for training (Teeling *et al.,* 2004b).

2.3 Sequence comparison-based methods for taxonomic assignment

Comparison-based methods rely on homology information obtained by database searches, e.g. using search tools like BLAST (Altschul *et al.*, 1990), to classify sequences based on the distribution of BLAST hits of predicted genes to taxonomic classes.

Huson and colleagues (Huson *et al.*, 2007; 2009) developed MEGAN, a method to explore the taxonomic content of a given metagenomic dataset. In a preliminary step, a set of sequence reference databases of choice, e.g NCBI-nr, NCBI-nt, NCBI-env-nr, or NCBI-env-nt (Benson *et al.,* 2006), are pooled. Raw sequence reads are then compared against these, using a similarity searching program such as BLAST. Subsequent analyses are then carried out using MEGAN, which employs a simple lowest common ancestor algorithm (LCA) to assign a read to its lowest common ancestor, using the NCBI taxonomy. In so doing, widely conserved sequences are assigned to higher order taxa closer to the root as opposed to species-specific sequences which are assigned at the leaves (Huson *et al.*, 2007). MEGAN was validated on the metagenomic dataset from the Sargasso Sea. The analysis attained a result that is in agreement with Venter *et al.,* (2004). In the same way, the mammoth bone dataset (Poinar *et al.*, 2006) was re-analyzed using MEGAN. Nearly 50% of the analyzed sequences were mapped to mammoth DNA, whereas the remaining sequences were found to be derived from endogenous bacteria and non-elephantid environmental contaminants (Poinar *et al.*, 2006), a result that is in agreement with what was obtained from MEGAN.

The CARMA algorithm developed by Krause *et al.*, (2008), relies on conserved domains and protein families (Finn *et al.*, 2006), to be able to predict the source organisms of given environmental DNA sequences. Using raw sequence reads as input, the algorithm searches for Pfam domains and protein family fragments (environmental gene tags- EGTs) that are conserved in the sample, by employing Pfam's profile hidden Markov models. In the second phase, the algorithm reconstructs a phylogenetic tree for each matching Pfam family. These trees consist of all previously detected EGTs and all other family members with known taxonomic affiliations. Using this method, it was observed that EGTs shorter than 30 amino acids could reliably be classified (Krause *et al.*, 2008), although at a high computational cost (Diaz *et al.*, 2009; Krause *et al.*, 2008).

Another method, Phymm with its hybrid PhymmBL, (Brady and Salzberg, 2009) incorporates interpolated Markov models (IMMs) in the process of characterizing unknown environmental gene sequences into definite phylogenetic groupings, based on information from multiple oligonucleotides of different lengths. Previously, IMMs were successfully applied to bacterial gene finding in the Glimmer system (Salzberg *et al.*, 1998), but had never been used for taxonomic characterisation of samples of unknown origin (Brady and Salzberg, 2009). Results from the acid mine drainage metagenome study (Tyson *et al.*, 2004), demonstrated that for short reads, Phymm as compared to previous methods such as PhyloPythia, showed a marked improvement in accurately classifying unknown fragments as short as 100 bp. Results from the hybrid method (PhymmBL) which incorporates information from both Phymm and BLAST, showed that this hybrid method outperforms either of the two single methods (Brady and Salzberg, 2009).

The SOrt-ITEMS algorithm developed by Monzoorul and colleagues (Monzoorul *et al.*, 2009) incorporates an orthology based approach, in addition to BLAST alignment parameters such as bitscore, alignment length, percent identity, to subsequently arrive at an appropriate level in the taxonomic tree at which a sequence read can be assigned. For hits that show significant orthology to a query read sequence, a reciprocal similarity search with the query read sequence (Monzoorul *et al.*, 2009) is carried out. All the hits identified as orthologs of the query sequence are then examined for the final assignment of the read. SOrt-ITEMS, which employs a modification of the algorithm used in MEGAN, shows improved taxonomic assignment over MEGAN, although it is more time consuming.

PPP utilises a MRCA approach which relies on sequence similarity search by BLAST. This approach is similar to the MRCA algorithm used in MEGAN and SOrt-ITEMS; the specifics of PPP are detailed below.

2.4 The Pathogen Profiling Pipeline (PPP)

PPP is an integrated system for rapidly surveying the microbial population in complex template mixtures without the need for laboratory cultivation. In a pre-processing step, nucleic acids are first extracted from representative clinical specimens using established laboratory protocols followed by Roche GS FLX pyrosequencing of random shotgun metagenomic libraries. This provides input sequence data for the PPP. This sequence data is then surveyed for microbial sequence signatures in a flexible manner. Researchers can construct a custom data analysis pipeline using the web interface by defining sets of reference databases to which input sequence reads are compared and the order in which they are to be searched. The sequence reads are compared to the reference databases (referred to as "steps") using BLAST alignment algorithms, then filtered according to thresholds (e.g. alignment length, percent identity). Using adjustable cut-off values of High scoring Segment Pair (HSP) length and percent identity, high scoring reads are assigned as “hits” to the step and subtracted from downstream analysis. A taxonomic rank filter is also provided where a maximum taxon rank is applied from a read (Pertsemlidis and Fondon, 2001). A concept of 'equivalent hits' is used and these are BLAST results falling within an assigned percentage of the top hit's bit score. The researcher may assign this cut-off during pipeline construction. Equivalent hits are used to taxonomically assign the sequence read using a simple MRCA approach. Reads lacking hits or that fail threshold filtering are optionally recombined with unassigned reads from other analysis steps, and then passed along to subsequent steps within the analysis pipeline. The unfiltered reads progress to subsequent filters for pathogen identification (e.g. viruses, bacteria, parasites, fungi). This strategy is useful for removing sequence reads derived from host, and helps to organize the large, complex set of input reads into logical subsets for additional downstream analyses. Reports are generated for each filtering step that detail the sequence match quality, taxonomic assignment, hits to similar reference sequences, and read abundances at different levels of taxonomic specificity. Reads and reports are also exportable for additional analyses in other metagenomics applications. For maximum throughput within tight time constraints (such as in emergency response situations), the PPP application is run on a high performance parallel computing cluster to distribute the compute-intensive analyses (Matthews *et al.*, unpublished).

It is highly desirable that all sequence reads obtained from a metagenomic microbial survey are assigned to a particular strain in the community. However, this is not usually possible due to the differences in abundance of the strains and variation of sequence coverage. The MRCA approach used in the PPP fails to account for this property in the way reads are assigned to their source organisms on several fronts. It fails to account for possible confounding factors such as sequence complexity and the degree of representation of identical sequences within other microbes. PPP also fails to consider taxonomic assignments within the context of the larger overall read data set, wherein assignments may be supported by other (very numerous) read taxonomic assignments. Figure 1 illustrates the workflow in a typical analysis pipeline in the PPP.

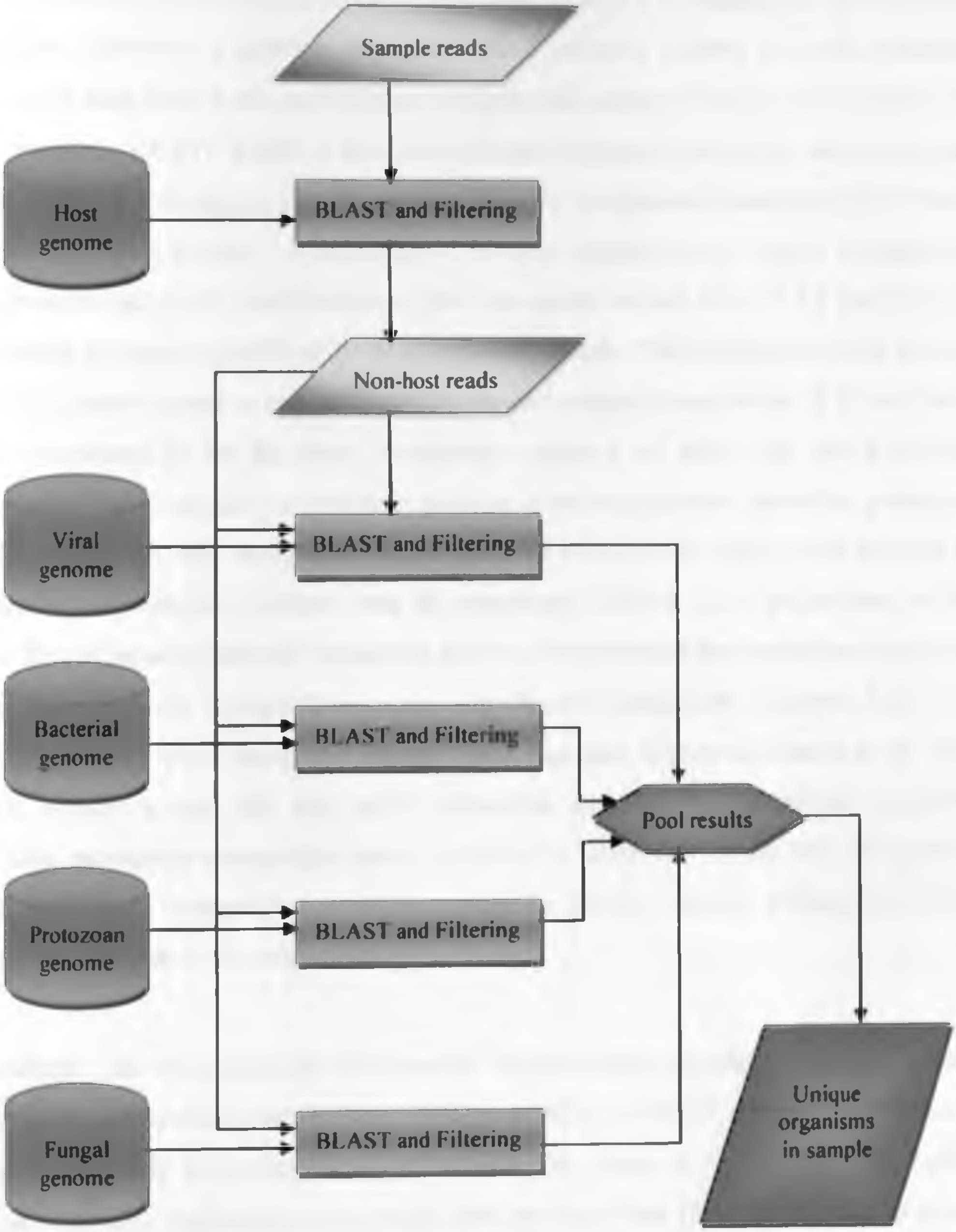


Figure 1: *Flowchart illustrating a typical analysis pipeline*

2.5 Summary

Assigning a taxonomic classification to all the sequence reads in a metagenomic dataset remains a difficult task. However, sequence-comparison based methods achieve this with reasonable accuracy and as such have been used in most metagenomic studies (Wooley *et al.*, 2010). This approach involves a BLAST search of the environmental (metagenomic) query sequence against a reference database. Filtering by significance of similarity is achieved by analysing the bit score, E-value or the percent identity. A taxonomy id is then assigned to the query environmental sequence based on taxonomic identification of the corresponding best BLAST hit thus providing a corresponding taxonomic profile of the environmental sample. The advantage of this approach is the use of all known genes as reference and the modest computational effort (Rolf and Carola, 2009). This appeared to be the most appropriate approach to adopt for this study since metagenomic samples constitute a diverse population of microorganisms including prokaryotes and viruses. Due to this fact, composition-based methods would not be appropriate because they do not reliably differentiate between taxa for sequences derived from prokaryotes or viral sequences. This is because bacterial sequences are so closely related that properties such as GC content or tetranucleotide composition cannot significantly distinguish between them at the species or strain level when using short (under 1 kb) sequence fragments (Kunin *et al.*, 2008). Conversely, viruses co-opt the host cell's replication machinery to replicate themselves; therefore, their nucleotide composition tends to evolve to match that of the host (Aragone's *et al.*, 2010). Nucleotide composition therefore cannot be used to reliably distinguish different viruses that have tropism to the same host.

All the methods that taxonomically characterise metagenomics samples based on sequence composition i.e Phylopythia, CompostBin, TETRA and SOM, s-GSOM were not considered for evaluation in this study as explained above. Additionally, some of the methods for example PhyloPythia were only applicable to sequences that are more than 1kb (1000 bp) yet the reads obtained from GS FLX experiments are much shorter (~250-500 bp).

Conversely, Phymm and PhymmBL, (Brady and Salzberg, 2009) which is based on sequence comparison was not evaluated. This was because the algorithm implemented in the PhymmBL program could not be used for taxonomic characterisation of samples of unknown origin (Brady and Salzberg, 2009). The SOrt-ITEMS[1] software (Monzoorul *et al.*, 2009) and the CARMA[2] software (Krause *et al.*, 2008) were too not considered for evaluation.

[1] The SOrt-ITEMS software obtained from http://metagenomics.atc.tcs.com/binning/SOrt-ITEMS/ was installed on the workstation. During testing, several bugs were reported by the developer and the fix is still in progress.

[2] Version 1.2 of the stand alone CARMA software was obtained from http://www.cebitec.uni-bielefeld.de/brf/carma/, but the program could not be installed because of its dependence on an outdated operating system environment that was no longer available.

CHAPTER THREE:METHODOLOGY

3.1 Introduction

This chapter presents the data and methods of analysis that were adopted to achieve the objectives of this study. PPP was downloaded and installed on the workstation. MEGAN was also downloaded and installed on the workstation along with its dependencies. The QSA Read simulator application was obtained as Perl scripts and put in a location on the workstation where it could be accessed. Simulated test data sets were generated by the QSA Read simulator and used to evaluate both the MRCA and the MEGAN algorithms on the basis of classification accuracy. The results from both tests were compared and a novel weighted MRCA algorithm developed. The new algorithm was evaluated on the same simulated dataset in terms of classification accuracy and later implemented in the PPP alongside the original MRCA. The workflow is illustrated in the flowchart (Figure 2).

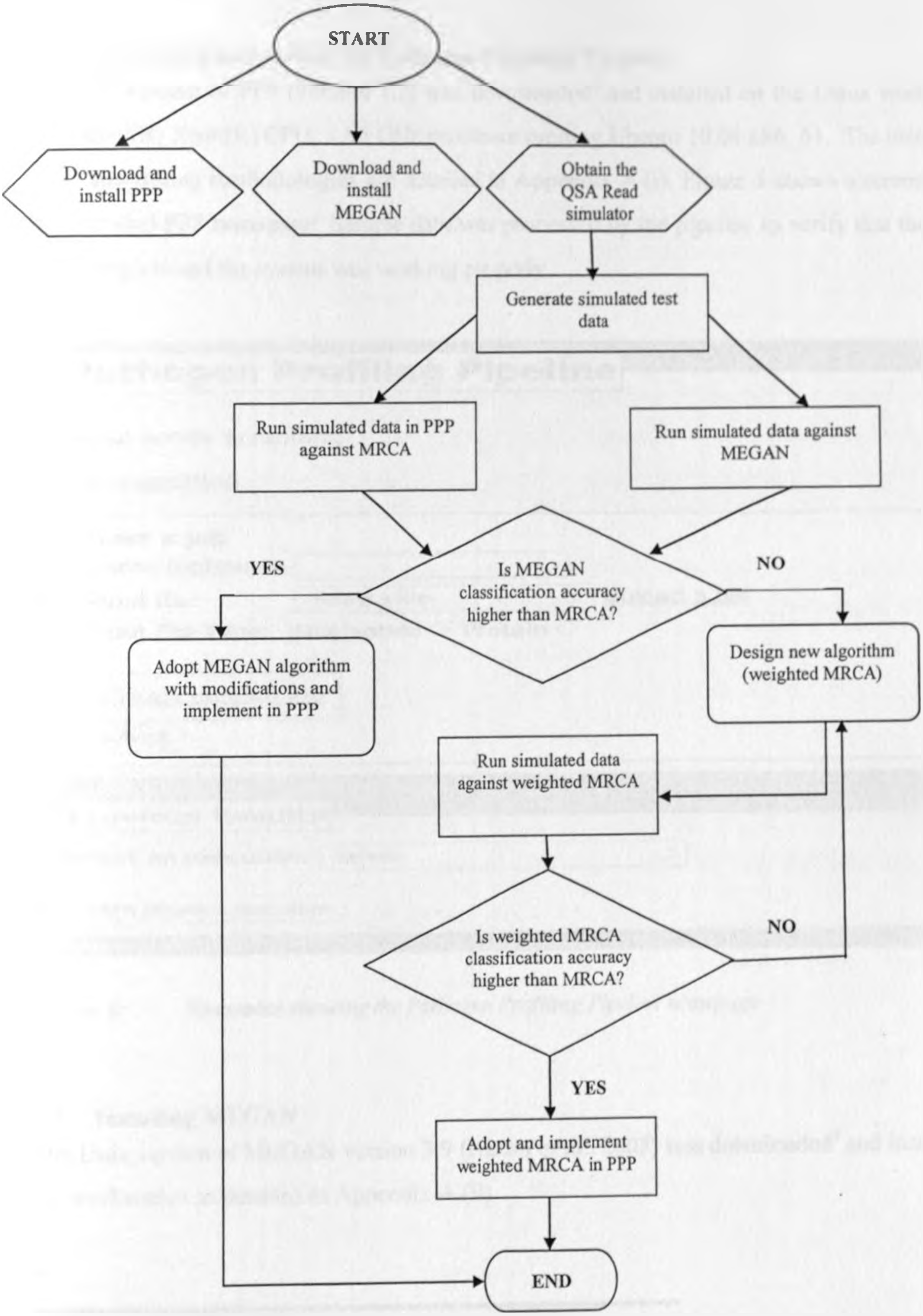
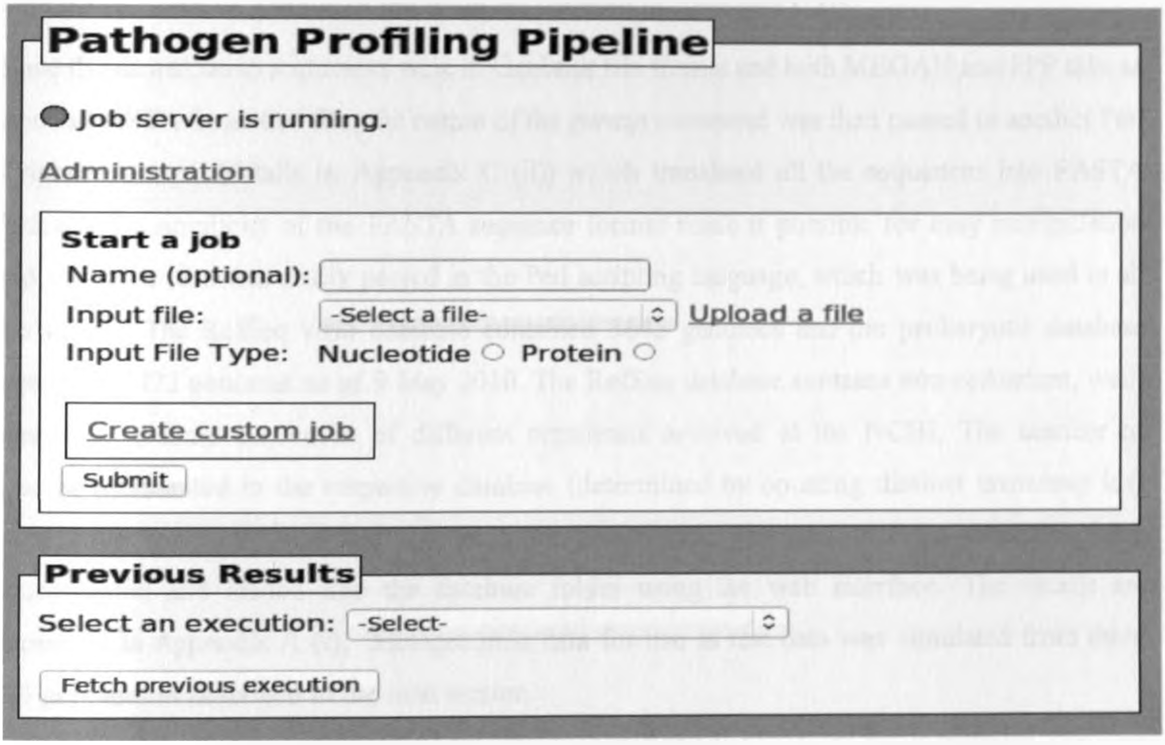

START
Download and install PPP
Download and install MEGAN
Obtain the QSA Read simulator
Generate simulated test data
Run simulated data in PPP against MRCA
Run simulated data against MEGAN
Is MEGAN classification accuracy higher than MRCA?
YES
NO
Adopt MEGAN algorithm with modifications and implement in PPP
Design new algorithm (weighted MRCA)
Run simulated data against weighted MRCA
Is weighted MRCA classification accuracy higher than MRCA?
NO
YES
Adopt and implement weighted MRCA in PPP
END

Figure 2: *Flowchart illustrating the project workflow*

3.2 Installing and testing the Pathogen Profiling Pipeline

A stable version of PPP (version 1.2) was downloaded[3] and installed on the Linux workstation with Intel(R) Xeon(R) CPU, 1.86 GHz processor running Ubuntu 10.04 x86_64. The installation steps and testing methodologies are detailed in Appendix A (i). Figure 3 shows a screenshot of the installed PPP homepage. Sample data was processed by the pipeline to verify that the install was complete and the system was working properly.



Pathogen Profiling Pipeline

● Job server is running.

Administration

Start a job

Name (optional):

Input file: -Select a file- Upload a file

Input File Type: Nucleotide ○ Protein ○

Create custom job

Submit

Previous Results

Select an execution: -Select-

Fetch previous execution

Figure 3: *Screenshot showing the Pathogen Profiling Pipeline homepage*

3.3 Installing MEGAN

The Unix version of MEGAN version 3.9 (Huson *et al*., 2007) was downloaded[4] and installed on the workstation as detailed in Appendix A (ii).

[3] http://www.corefacility.ca/ppp/

[4] http://www-ab.informatik.uni-tuebingen.de/data/software/megan/

3.4 Obtaining data

Full prokaryotic (Bacteria and Archaea) and viral genomes were downloaded[5] from the NCBI RefSeq database, release 41. The Perl script, *dl.pl* was written and used to dynamically download the databases. The script takes as input the FTP address above for either prokaryotic or viral database, downloads all the different segments of the database in question and subsequently unzips them using a system command executed from the same directory where the databases were downloaded to. Details of the script are outlined in Appendix C (i).

Since the downloaded sequences were in Genbank file format and both MEGAN and PPP take as input a FASTA formatted file, the output of the *gunzip* command was then passed to another Perl script *convert.pl,* (details in Appendix C (ii)) which translated all the sequences into FASTA format. The simplicity of the FASTA sequence format made it possible for easy manipulation and sequence data was easily parsed in the Perl scripting language, which was being used in all the scripts. The RefSeq viral database contained 3642 genomes and the prokaryotic database contained 3473 genomes as of 9 May 2010. The RefSeq database contains non-redundant, well-annotated genome sequences of different organisms archived at the NCBI. The number of species represented in the respective database (determined by counting distinct taxonomy ids) were 2506 for viral genomes, and 5458 for prokaryotic genomes. All the databases were concatenated and loaded into the database folder using the web interface. The details are appended in Appendix A (i). Metagenomic data for use as test data was simulated from these full genomes as described in the next section.

[5] ftp://ftp.ncbi.nih.gov/refseq/release/41/

3.5 The QSA Read simulator

In order to carry out test analyses, simulated test data was required. This dataset was created using the QSA Read simulator[6]. The application written in Perl simulates sequence reads and errors as would be found in a pyrosequencing data set.

3.5.1 Simulation of test data sets

QSA Read simulator takes a FASTA file of known genome sequences or contigs as input, then randomly generates simulated reads from the input. To make the generated reads more authentic, the tool applies errors to the reads based on pyrosequencing error models (Huse *et al.*, 2007).

The tool's processing pipeline consists of several phases:

- Selection of source genome sequences from the internal database
- Configuration of the species abundance profile by setting the relative copy number of the genome sequences
- Application of technology-specific error models to the fragments to create sequencing reads

i) Selection of source genome sequences

The source organisms used consisted of all full prokaryotic and viral genome sequences which were downloaded from the Refseq database at the NCBI.

ii) Configuration of Species Abundance Profiles

Full prokaryotic and viral genome sequences downloaded from Refseq database were stored locally as source sequences thus constituting a local integrated database. The relative abundances (numeric) of each genome sequence were specified in a text-based profile file by counting the distinct instances of each genome sequence.

[6] QSA Read simulator obtained from Tom Matthews at NML, PHAC

iii) Application of technology-specific error models to the fragments to create sequencing reads

a) Read sampling

For the simulation of read sequences, statistical approaches in Huse *et al.*, 2007 were adopted to simulate the distribution of read lengths, the frequency rate and the use of pyrosequencing error models. Large fragments (clones) with a length of 400 bp and a standard deviation of 10 bp were modelled with a normal distribution N(400,10).

b) Simulation of pyrosequencing reads

The intensity of emitted light was used to estimate the length of homopolymers, i.e. runs of identical nucleotides in a sequence.

Let **r** denote the length of a given homopolymer. The intensity of emitted light was modelled using a normal distribution $\mathbf{N(\mu,\sigma)}$, with mean $\mu = \mathbf{r}$ and standard deviation $\sigma = \mathbf{k.\sqrt{r}}$, where **k** is a fixed proportionality factor. Following Huse *et al.*, 2007, by default k = 0.15 was used. Although basic statistics imply that the standard deviation should grow with the square root of r, in Huse *et al.*, 2007, the standard deviation of the light intensity emitted during 454-sequencing is reported to be $\sigma = \mathbf{k.r}$. Both variants of the calculation were implemented in the software.

A *negative flow* is a flow of nucleotides in which the sequence to synthesize is not elongated. Light intensities of negative flows follow a lognormal distribution, with mean $\mu = 0.23$, and standard deviation $\sigma = 0.15$, (Huse *et al.*, 2007). A random variable X is said to be lognormally distributed, if the random variable ***ln*(X)** is normally distributed.

Base-calling intensities of negative flows were simulated and the misinterpretation of null-mers were modelled as homopolymers of length = 1 (insertion). The algorithm takes the order of the sequencing flows into account. Since the nucleotides are cyclically flowed in the order T,A,C,G, after a given base only two specific negative flows in a specific order were allowed.

c) Applying empirical models to simulated reads

The simulator includes an empirical error model that allows the incorporation of user-defined error statistics. The general approach as described in Engle *et al.*, 1994, in which the probability of an occurrence of a sequencing error depends on the position of the erroneous base and its surrounding bases, was taken.

The error model used in QSA Read simulator was based on mappings (error curves) that assign error rates to specific base positions. Each mapping has three parameters (the last two are optional):

i. type of error (deletion, insertion, substituion),
ii. base at the position where the error occurs and
iii. base preceding the position where the error occurs.

The QSA Read simulator application was designed to be run on a modern Linux platform and required the following packages:

- Perl
- BioPerl, both of which were already installed.

The directory containing the application was obtained and placed in the applications folder, on a location on the computer from which the application was run. The QSA Read Simulator was initiated as follows. Other details are described in Appendix A (iii).

```
$ cd ~/apps/QSAreadsim
```

The following command was issued to simulate 10000 sequence reads from the RefSeq microbial database FASTA file (refmicrobial.fna), of average length 206 within a standard deviation of 10 and with an accession prefix of "RM". The error rates associated with this simulated pyrosequencing data are incorporated in the QSA Read simulator.

```
$ ./readsim.pl -r /home/hellen/apps/data/refmicrobial.fna -l 206 -d 10
-i RM -n 10000 -o simrefmicrobial_microbial_reads.fna
```

For the simulated viral data set, both MRCA and the MEGAN algorithm were evaluated on a simulated metagenome consisting 10000 reads from 196 complete genomes. These were simulated on the basis of the GS FLX pyrosequencing model with lengths 162-251 nucleotides and average length of 206 nucleotides. This constituted ≈ 5% of the database sequences.

In the simulation experiment consisting of prokaryotic simulated data, all the algorithms were evaluated on a metagenome consisting of 10000 reads from 2018 complete genomes. The created metagenome represented a complex microbial community, with sequence fragments from both Archaea and Bacteria.

The simulated data bears close similarity to a pyrosequencing experimental laboratory dataset since all the variables (mate-pairs, mutations, read lengths and relative abundances) that affect the sequencing process were incorporated in the QSA Read simulator program, that was used to generate the test data sets.

3.6 Testing MRCA (within PPP) and MEGAN using simulated test data

The data simulated above was loaded into the PPP using the web-based administration interface. This input set was searched for similarity using the BLAST program (BLASTN) implemented within the pipeline as a selectable option against the same RefSeq database (RefSeq microbial) for prokaryotic simulated data and Refseq viral database for viral simulated data.

For testing MEGAN, a stand-alone BLAST search (BLASTN) was run locally on the simulated sequences (same data as was used in PPP runs) and the output of the BLAST search used as input to MEGAN. The database searched against was the original RefSeq database (microbial for prokaryotic genomes and viral for the viral genomes). This allowed for determining which of the sequences were falsely classified into a given class, since the correct taxonomic label was already known.

```
$ blastall -p blastn -d ~/apps/db/microbial.fna -i
~/apps/data/simrefmicrobial_microbial_reads.fna
```

The output of BLASTN was thereafter used as input to MEGAN using a bit score threshold of 40 (same as in the PPP) and retaining only those hits that are within 10% of the top hit for each read.

Both PPP and MEGAN output files were analysed using Perl scripts to determine the classification accuracy of each algorithm in assigning metagenomic reads to the respective taxonomic labels. The Perl script ***ranks_filter.pl*** (Appendix C (iii)) takes as input the output of a PPP run (hits file) in a FASTA format and cross-references it with the original database to determine if the assignments obtained by the MRCA algorithm in PPP, are the same as those in the reference database that the simulated sequences were "BLASTed" against. The script then returns the abundances (as a text file) of sequences that were correctly mapped to their source organisms. If the sequence was mapped incorrectly, it returned the taxonomic rank at which that read sequence was assigned.

MEGAN returned the phylogenetic diversity of a given dataset, along with the abundances obtained for each taxonomic class. The taxonomic assignments obtained from both applications were analysed for sensitivity, specificity and accuracy.

3.7 Accuracy comparisons for MRCA (PPP) and MEGAN

By comparing the predicted taxa with the known taxa at each of the taxonomic levels, the sensitivity, specificity and accuracy of both algorithms were assessed as follows.

For a given taxonomic class *c*, true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN) were obtained as explained below;

TP_c -number of reads that were correctly classified into class *c*

FP_c -number of reads that were erroneously assigned to class *c*

FN_c -number of reads from class *c* that were misclassified into some other class $d \neq c$

For the RefSeq database, all the generated reads essentially belonged to a class in the database, thus all those that did not return any hits in a pipeline run were classified as false negatives

TN_c - number of reads that were correctly classified as not belonging to class *c*

For each of the datasets, false negatives and true negatives were generated by removing the sequences corresponding to the input reads from each of the representative databases. The Perl script ***pseudo_db.pl*** (Appendix C (iv)) was written and used to construct a decoy database. The script takes as input the original input read sequences (from QSA Read Simulator), and compares these against the database from which they were generated. The script then writes out the sequences in the reference database that were not found in the input set. The sequences are formatted for BLAST analysis and uploaded to the databases folder. This new set of sequences is then treated as the new reference database against which the original input reads are searched for similarity. The viral database (decoy virus database) contained a total of 3533 full genome sequences after the filtering whereas the prokaryotic database (decoy microbial database) contained 2583 genomic sequences. False positives corresponded to the reads that were returned as hits (since the source organism sequences had actually been removed from the database) at the different taxonomic levels as opposed to true negatives which were the number of reads that remained unassigned (lacking from the database, which is expected).

The sensitivity, specificity and accuracy for each of the algorithms were assessed as follows; Sensitivity measures the proportion of reads that is correctly classified and for a given taxonomic class. It is given by;

$$\textbf{\textit{Sensitivity}} = TP_c / (TP_c+FN_c)$$

Specificity on the other hand measures the reliability of classifications and is defined as;

$$\textbf{\textit{Specificity}} = TN_c / (TN_c+ FP_c)$$

The accuracy of a given measurement system is the degree of closeness of measurements of a quantity to its actual (true) value. Accuracy is determined by comparing the measurement against the true or accepted value and is given by;

$$\textbf{\textit{Accuracy}} = TP_c + TN_c / (TP_c + FP_c + FN_c + TN_c)$$

The performance was evaluated at each of the taxonomic levels of species, genus, up to family level.

In order to validate the results obtained from the classification accuracy* tests, a nonparametric two-tailed z-test (α = 0.05) was run in SPSS 14.0. In all the cases, two hypotheses were formulated as follows;

Hypotheses:

Null hypothesis (H0): p1-p2 =0

Alternative hypothesis (H1): p1-p2 $\neq$ 0

Where p1 is the classification accuracy of MEGAN

p2 is the classification accuracy of MRCA

Rejection region:

The null hypothesis would be rejected if *p*-value $\leq$ 0.05.

*Encompasses all the tests i.e sensitivity, specificity and accuracy.

3.8 The weighted MRCA algorithm

A novel weighted MRCA approach was developed as a number of Perl scripts. The developed method was tested against the existing MRCA algorithm in the PPP. During development of the weighted MRCA algorithm, the motivation was to improve on the accuracy of taxonomic assignment relative to the existing MRCA algorithm. The weighted MRCA strategy utilizes the breadth-first search algorithm. Following a BLAST search of the input reads against the database of interest, all the equivalent hits to each of the query input reads were examined and a rooted taxonomic tree constructed containing the taxonomic assignment for each read. Each terminal node in the tree is assigned the bit score from the BLAST result. Each parent node in the tree is assigned the sum of the bit scores of its immediate children. A breadth-first search of the tree is then performed, from root to tip. The node containing the smallest bit score that is greater than or equal to a cut-off bit score (e.g. 2-thirds of the maximum cumulative bit score in the tree) is assigned the weighted MRCA for that dataset.

3.8.1 Optimization of bit-score cut-off for the weighted MRCA algorithm

The bit score cut-off was tested on both simulated datasets (viral and prokaryotic genomes) to determine at what value the algorithm returned the best overall taxonomic accuracy. This was done by running the algorithm with different values as bit score cut-off and noting the results. The percentage bit score value that returned the highest number of positively assigned reads was adopted as the bit score cut-off for subsequent runs.

3.9 Evaluation of weighted MRCA algorithm performance against unweighted MRCA

The sequence reads used to test the weighted MRCA were derived from the hits that were obtained from a pipeline run with the original MRCA, constituting only those that returned four or more equivalent hits. This is because if a query sequence read mapped to more than four 'equivalent hits' (i.e. hits passing a threshold percent identity and length cut-off) the taxonomic diversity present was sufficiently large to produce an estimated MRCA at ranks higher than genera (i.e. family, order, class, phylum and kingdom, whereas taxonomic trees constructed from four or fewer equivalent hits tended to yield MRCA predictions at the rank of genera or lower.

The Perl script ***get_accession.pl*** (Appendix C (v)) was written for the purpose of generating the reference genome dataset. The script takes as input an array of read ids from the PPP output and obtains their original accession numbers and sequence data from the respective RefSeq database. For each of the accession numbers, the corresponding sequence data was obtained and this was used as input to the QSA Read Simulator to construct a simulated dataset with the same parameters as described above in Section 3.4. The identity of the genome sequence from which each read was generated was recorded with the unique read id in order to assess the accuracy of the MRCA algorithms. The new dataset was run through the pipeline against both the MRCA and weighted MRCA algorithms. The results were analysed using the Perl script ***ranks_filter.pl*** and the sensitivity, specificity and accuracy are reported.

The nonparametric two-tailed z-test ($\alpha = 0.05$) was run in SPSS 14.0 in the same manner as previously reported. The two hypotheses were formulated as follows;

Hypotheses:

Null hypothesis (H0): $p1-p2 = 0$

Alternative hypothesis (H1): $p1-p2 \neq 0$

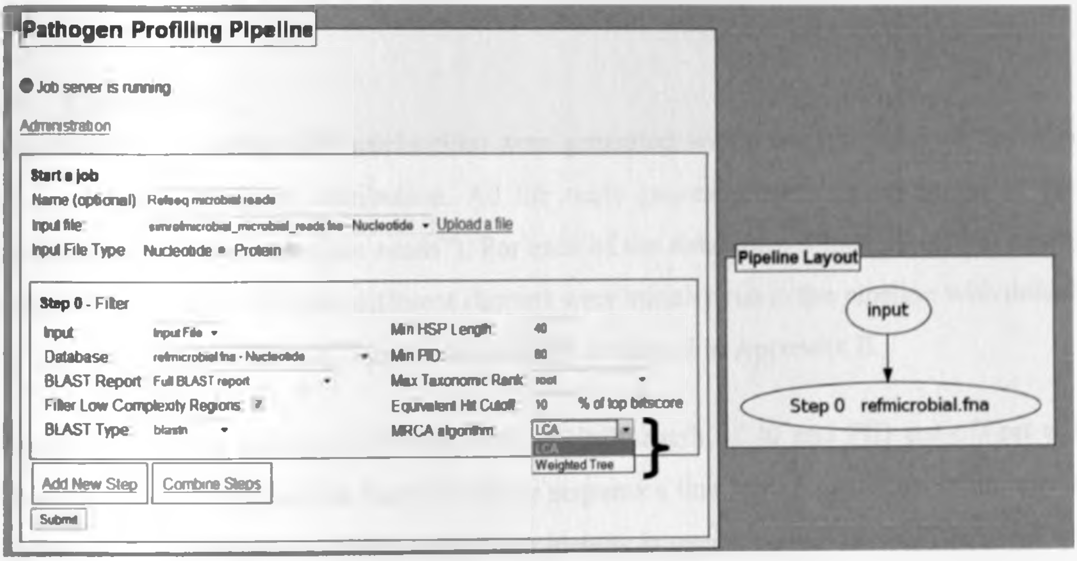
Where p1 is the classification accuracy of MRCA

p2 is the classification accuracy of weighted MRCA

Rejection region:

The null hypothesis would be rejected if p-value ≤ 0.05.

After testing, the weighted MRCA algorithm was implemented as a function (mrca) in the ***LCA.pm*** (Appendix C (vi)) module which was loaded in the *ppp-backend/lib* directory. A wrapper script *lcawrap.pl* (Appendix C (vii)) was written to analyse all of the "equivalent hits" that were obtained from a given pipeline run since the hits obtained from both the MRCA and the weighted MRCA were the same (both functions run the same BLAST algorithm, with the same parameters), but differ in the way the equivalent hits were treated. The weighted MRCA option was exported to the front-end (Figure 4) as a selectable option from a drop-down menu by incorporating it in the appropriate scripts in the */var/www/ppp-web/cgi-bin* directory.



Pathogen Profiling Pipeline

● Job server is running

Administration

Start a job

Name (optional) Refseq microbial reads

Input file: sim/refmicrobial_microbial_reads.fna - Nucleotide

Input File Type Nucleotide ● Protein

Step 0 - Filter

Input: Input File

Database: refmicrobial.fna - Nucleotide

BLAST Report: Full BLAST report

Filter Low Complexity Regions: ☑

BLAST Type: blastn

Min HSP Length: 40

Min PID: 80

Max Taxonomic Rank: root

Equivalent Hit Cutoff: 10 % of top bitscore

MRCA algorithm: LCA

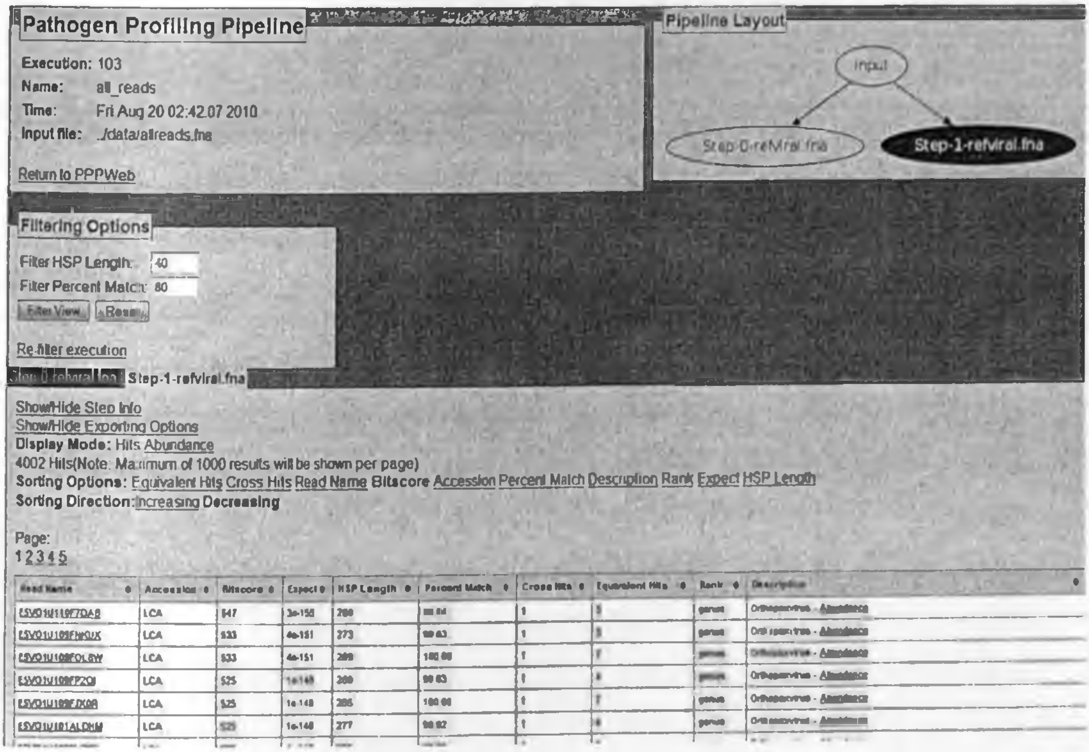
Figure 4: *The weighted MRCA algorithm implemented as a selectable option (right curly brace) from a drop-down menu alongside the original MRCA algorithm*

CHAPTER FOUR: RESULTS AND DISCUSSION

4.1 RESULTS

Read lengths of average 206 nucleotides were generated within a given standard deviation (10 nt) to enable a uniform distribution. All the reads generated were in the range of 162-251 nucleotides ("random shot-gun reads"). For each of the databases, a total of 10,000 reads were randomly generated. The two different datasets were initially run in the pipeline with default cut-off values (PID 80, HSP 40). Sample datasets are presented in Appendix B.

Figure 5 shows an example PPP run with an HSP length of 40 and PID cut-off set at 80 as filtering options. This implies that only those sequences that have a similarity spanning over 40 nucleotide bp and those for which the percent identity is over 80% will be considered as "hits" to the query sequence read in question.



Pathogen Profiling Pipeline

Execution: 103
Name: all_reads
Time: Fri Aug 20 02:42:07 2010
Input file: ./data/allreads.fna

Return to PPPWeb

Pipeline Layout

Filtering Options

Filter HSP Length: 40
Filter Percent Match: 80

Re-filter execution

Show/Hide Step Info
Show/Hide Exporting Options
Display Mode: Hits Abundance
4002 Hits(Note: Maximum of 1000 results will be shown per page)
Sorting Options: Equivalent Hits Cross Hits Read Name Bitscore Accession Percent Match Description Rank Expect HSP Length
Sorting Direction: Increasing Decreasing

Page:
1 2 3 4 5

Read Name	Accession	Bitscore	Expect	HSP Length	Percent Match	Cross Hits	Equivalent Hits	Rank	Description
ESVO1U110F7DAB	LCA	547	3e-155	280	[illegible]	1	5	genus	Orthopoxvirus - Abundance
ESVO1U109FNKUX	LCA	533	4e-151	273	99.63	1	5	genus	Orthopoxvirus - Abundance
ESVO1U109FOL8W	LCA	533	4e-151	289	100.00	1	7	genus	Orthopoxvirus - Abundance
ESVO1U109FP2QI	LCA	525	1e-148	280	99.63	1	8	genus	Orthopoxvirus - Abundance
ESVO1U109FJX0R	LCA	525	1e-148	285	100.00	1	7	genus	Orthopoxvirus - Abundance
ESVO1U101ALDHM	LCA	525	1e-148	277	99.82	1	8	genus	Orthopoxvirus - Abundance

Figure 5: *Screenshot of the PPP showing an example pipeline run.*

Following a BLAST search and additional taxonomic classification, the MRCA was obtained in the example above as "*Orthopoxvirus*" (from the description tab). This taxonomic assignment is at a *genus* rank which is an informative label, although a more specific assignment at a species rank would have been preferred. This however is a narrow taxonomic label and more laboratory experiments can be done as a follow-up to confirm which species in the *Orthopoxvirus* genus is the source organism from which the metagenomic dataset was obtained.

4.1.1 Accuracy comparison between MEGAN and MRCA algorithm

Table 1: Classification performance of MEGAN and MRCA for the simulated RefSeq prokaryotic database

	MEGAN			MRCA		
	Species	Genus	Family	Species	Genus	Family
Sensitivity (%)	82.3**	89.9**	95.4	83.3**	90.6**	97.5
Specificity (%)	95.2**	93.1**	89.7**	95.8**	94.3**	90.2**
Accuracy (%)	83.3	85.2	92.1**	87.4	93.6	96.5**

** *p*-values obtained were greater than p=0.05 (0.063-0.862)

More than 65% (indicated by asterisk) of the observed classification accuracies had *p*-values > $0.05 = \alpha$.

4.1.2 Testing the weighted MRCA algorithm

The developed weighted MRCA algorithm was tested on a simulated viral dataset and the illustration shows the taxonomic tree that was generated from the pipeline run. The different taxonomic ranks at which both the MRCA (original algorithm) and the weighted MRCA assigned the query dataset are indicated (with arrows) on the figure.

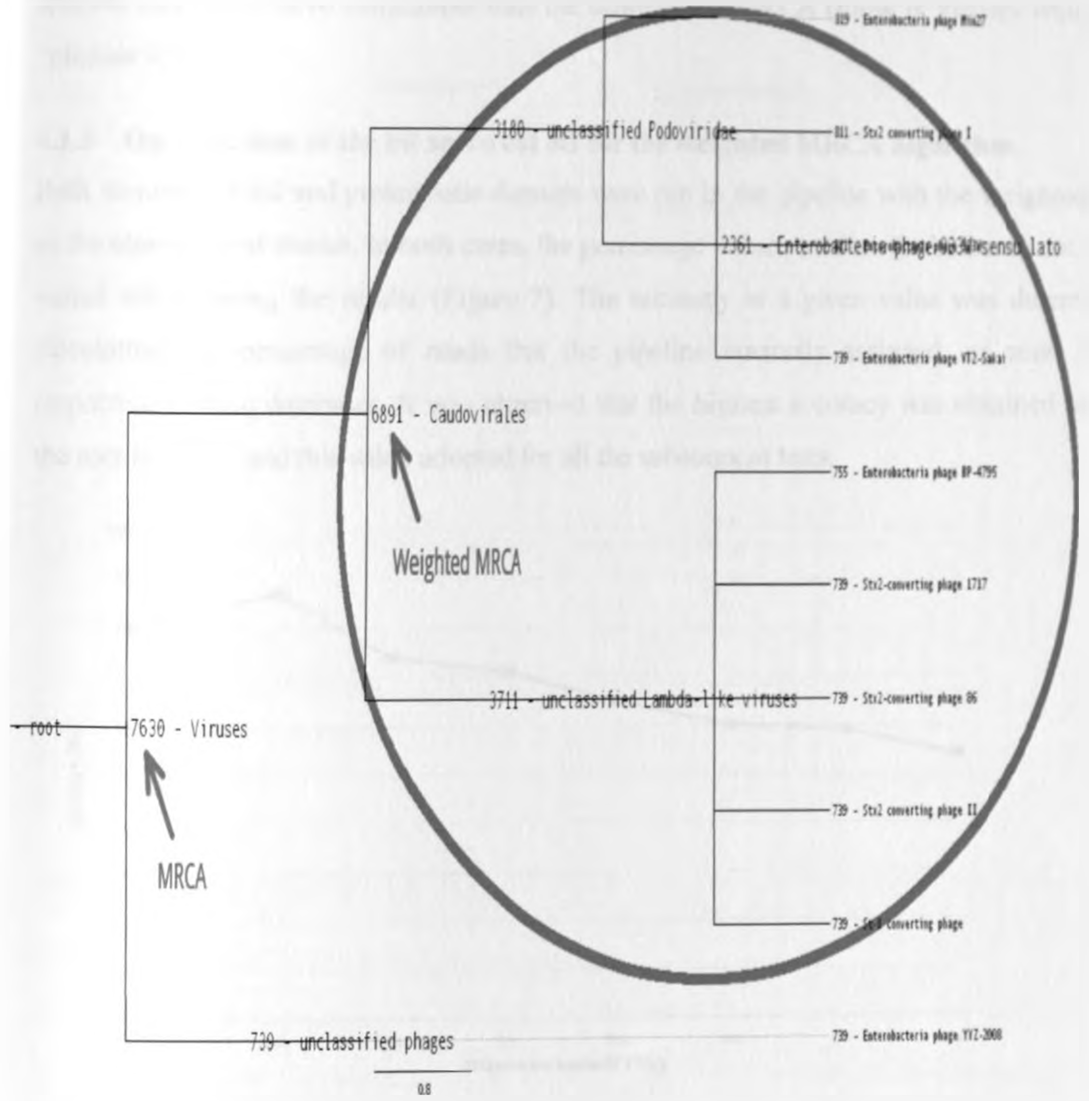

819 - Enterobacteria phage Min27
3180 - unclassified Podoviridae
811 - Stx2 converting phage I
2361 - Enterobacteria phage 933W sensu lato
739 - Enterobacteria phage VT2-Sakai
6891 - Caudovirales
755 - Enterobacteria phage BP-4795
Weighted MRCA
739 - Stx2-converting phage 1717
3711 - unclassified Lambda-like viruses
739 - Stx2-converting phage 86
root
7630 - Viruses
739 - Stx2 converting phage II
MRCA
739 - Stx1 converting phage
739 - unclassified phages
739 - Enterobacteria phage YYZ-2008
0.8

Figure 6: *Illustration of the weighted MRCA algorithm*

The numbers in the figure show example bit scores obtained from a pipeline BLAST search, the text shows the corresponding taxonomic assignment. In this example, the bit score cut-off was set at 6000 and the weighted MRCA algorithm classified the input metagenome as belonging to the family of *Caudovirales*, [the last node (descending from root to tip) with a bit score value that is above the cut-off, and one for which all the children's nodes have bit scores lower than the cut-off]. The original MRCA however returned *Viruses* as the taxonomic group to which the input belongs. In this case, the weighted MRCA gives the taxonomic rank as "Family", a more specific and informative assignment than the unweighted MRCA (Rank is Viruses which is at a "phylum level").

4.1.3 Optimization of the bit score cut off for the weighted MRCA algorithm

Both simulated viral and prokaryotic datasets were run in the pipeline with the weighted MRCA as the algorithm of choice. In both cases, the percentage values used as the bit score cut off were varied while noting the results (Figure 7). The accuracy at a given value was determined by calculating the percentage of reads that the pipeline correctly assigned, as seen from the respective RefSeq databases. It was observed that the highest accuracy was obtained at 60% of the root bit score and this value adopted for all the subsequent tests.

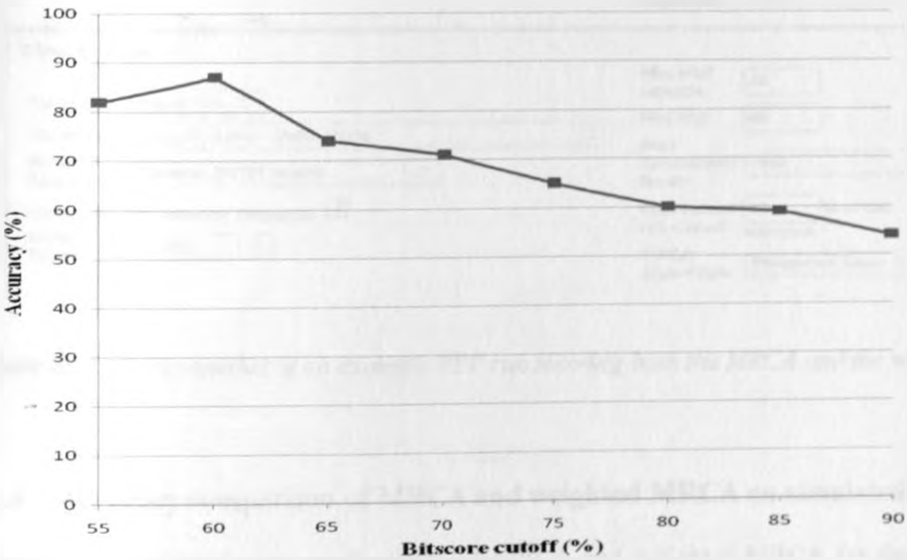
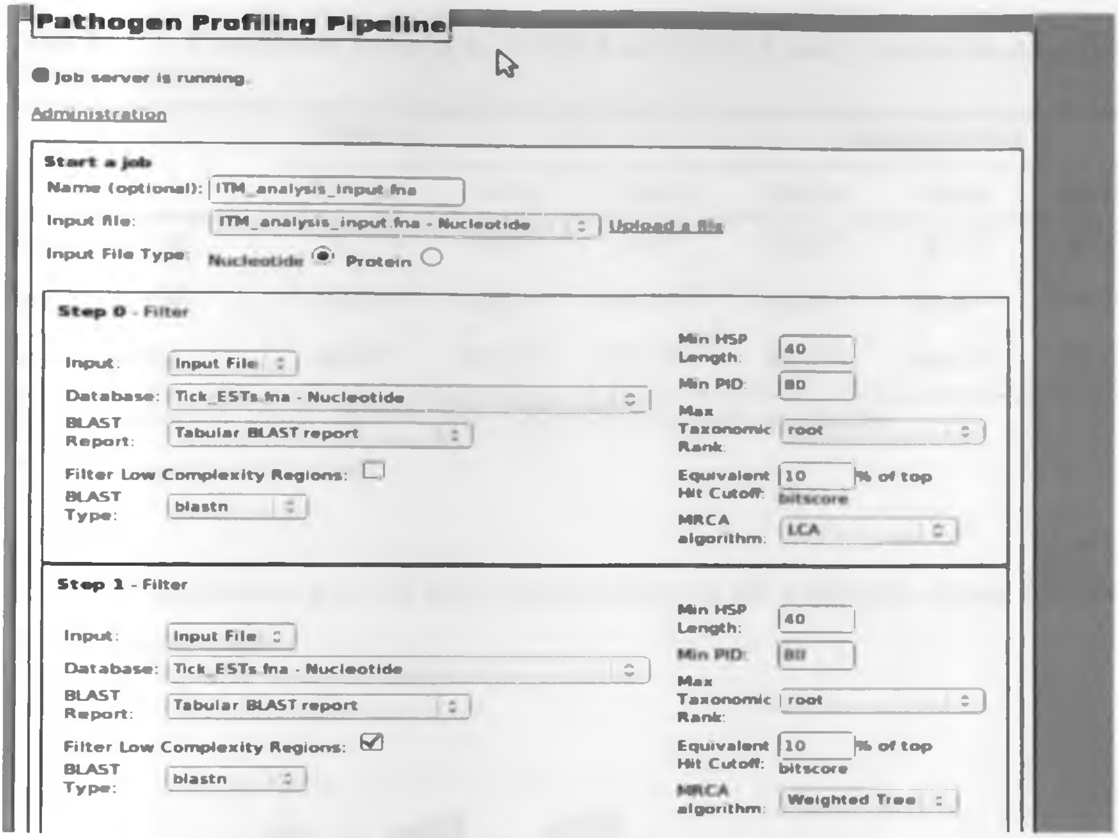


Figure 7: *Optimisation of the bit score cut off for the weighted MRCA algorithm*

Another example analysis in which both the MRCA and the weighted MRCA were employed in just one pipeline run (Step 0- MRCA, Step 1- weighted MRCA) is shown in Figure 8. In this case, both methods yielded the same number of hits but the equivalent hits for each of the sequence reads were different. The appropriate Perl scripts were used to extract the specific taxonomic assignments at each of the steps. The accuracy of these assignments was tabulated in the Table 2.



Pathogen Profiling Pipeline

● Job server is running.

Administration

Start a job

Name (optional): ITM_analysis_input.fna

Input file: ITM_analysis_input.fna - Nucleotide   Upload a file

Input File Type: Nucleotide ● Protein ○

Step 0 - Filter

Input: Input File

Database: Tick_ESTs.fna - Nucleotide

BLAST Report: Tabular BLAST report

Filter Low Complexity Regions: □

BLAST Type: blastn

Min HSP Length: 40

Min PID: 80

Max Taxonomic Rank: root

Equivalent Hit Cutoff: 10 % of top bitscore

MRCA algorithm: LCA

Step 1 - Filter

Input: Input File

Database: Tick_ESTs.fna - Nucleotide

BLAST Report: Tabular BLAST report

Filter Low Complexity Regions: ☑

BLAST Type: blastn

Min HSP Length: 40

Min PID: 80

Max Taxonomic Rank: root

Equivalent Hit Cutoff: 10 % of top bitscore

MRCA algorithm: Weighted Tree

Figure 8: *Screenshot of an example PPP run showing both the MRCA and the weighted MRCA*

4.1.4 Accuracy comparison of MRCA and weighted MRCA on simulated metagenomes

Table 2: Classification performance of MRCA and weighted MRCA for the simulated RefSeq viral database

	MRCA			Weighted MRCA		
	Species	Genus	Family	Species	Genus	Family
Sensitivity (%)	87.3‡‡	91.9‡‡	93.4**	88.3‡‡	92.5‡‡	95.5**
Specificity (%)	89.2**	90.3**	93.6**	92.8**	94.5**	97.1**
Accuracy (%)	88.0**	91.5**	93.1**	91.6**	93.8**	96.3**

‡‡ *p*-values were 0.045 (z=1.70) and 0.023 (z=1.99) for species and genus respectively (α= 0.05)

***p*-values < 0.05 (0.039-0.004)

Table 3: Classification performance of MRCA and weighted MRCA for the simulated RefSeq prokaryotic database

	MRCA			Weighted MRCA		
	Species	Genus	Family	Species	Genus	Family
Sensitivity (%)	92.2**	96.9‡‡	97.3‡‡	97.0**	97.2‡‡	97.4‡‡
Specificity (%)	78.6**	80.1**	85.4**	93.2**	95.8**	98.2**
Accuracy (%)	86.4**	91.0**	92.1**	94.8**	96.1**	98.0**

‡‡*p*-values were 0.049 (*z*= 1.65) and *p*=0.026 (*z*= 1.94) for genus and family respectively.

***p*-values ranged from 0.014-0.003.

At α= 0.05 significance level, all the *p*-values obtained for this prokaryotic dataset were below the significance level.

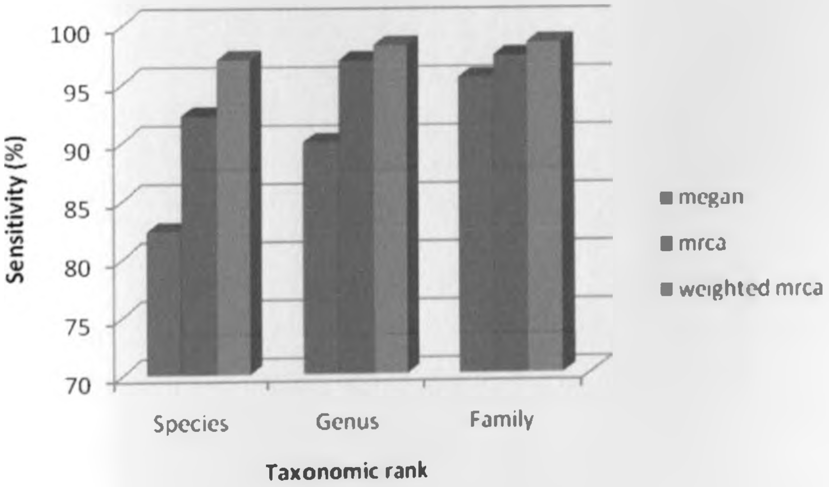


Figure 9: *The sensitivity of the three methods generated from the simulated prokaryotic dataset consisting 5458 distinct species from the NCBI Refseq database*

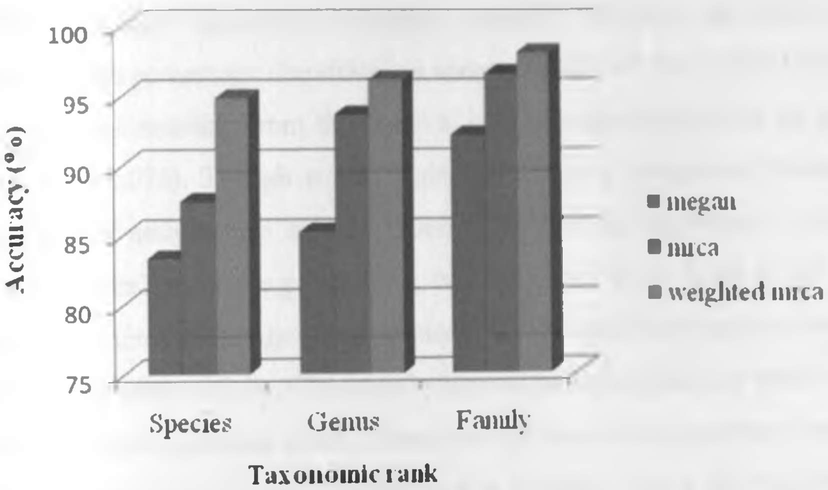


Figure 10: *The accuracy of the three methods generated from the simulated prokaryotic database of the NCBI Refseq database*

4.2 DISCUSSION

For the simulated prokaryotic database dataset, MEGAN and MRCA were comparable, achieving high percentage classification accuracy (82.3-97.5%). With MEGAN, the high rate of false negatives resulting from the removal of the species sequences, led to a low percentage in accuracy (*p*=0.076). Though it was logical to expect a progressive increase of the cumulative percentage of assignments as one moves from specific taxonomic ranks (species, genus) to higher taxonomic ranks (e.g. phylum), it is observed from Table 1 that both methods have a lower cumulative percentage of assignments at the family level as compared to that at the genus level in terms of specificity. This was because some species as they were being mapped up in the taxonomy tree, the naming scheme used was different from what was used at species level, thus some of these sequences would be returned as "no rank" in the analysis scripts. This contributes to a high number of TN in the sample dataset, as this is the main parameter that influences specificity. In future analyses, Perl scripts could be used to eliminate this inconsistency. These would extract the header sequence information then format all sequences in such a way that there is uniformity in all the headers.

From the same table, it was seen that more than 65% of the observed classification accuracies had *p*-values > 0.05 = α. This general trend was evidence that we do not reject the null hypothesis that the difference between the classification accuracy of MEGAN and that of MRCA is equal to zero.

The side-by-side analysis in which both the MRCA and the weighted MRCA are employed in the same pipeline run allows for minimising of computational time and resources, which is usually a bottleneck in most metagenomic analyses. The idea of a "processing pipeline" allows for making choices about which "steps" to be carried out and which programs to use (Wu *et al.*, 2008). Sequence comparison based approaches typically make use of the best BLAST hit obtained from a BLAST search of the environmental query sequence against a reference database. Filtering by significance of similarity for example based on percent identity, bit score or E-value leaves only the subject sequences that are more related to the query environmental sequence. A taxonomy id assigned to the query based on the taxonomic classification of the corresponding best BLAST hit thus provides a corresponding taxonomic profile of the environmental sample. This approach has several advantages including the low computational effort (if the environmental query dataset is not too large), and the use of all known genes as reference (from the RefSeq database in this case) (Bork *et al.*, 2007)

There was an improvement in taxonomic assignment at all ranks between MRCA and weighted MRCA algorithms for the viral simulated dataset. This is seen from Table 2 where the sensitivity values for the weighted MRCA algorithm are higher than those for the MRCA algorithm. The assignment of reads to a specific sub group within the bit score cut-off significantly contributes to this scenario. In some cases, the *p*-values are as low as 0.004 at species level which demonstrates that the two methods are significantly different (at a significance level of α= 0.05) in taxonomic classification performance. The reliability of the assignments is equally high as can be seen from the specificity values.

The accuracy with which the weighted MRCA assigns the viral query reads to the various taxa is higher than that for the MRCA algorithm. This is attributed to the fact that the algorithm restricts taxonomic assignment to levels of the sub tree where in all the children nodes are above a certain cut off. This ensures that the more general assignments, as long they have fewer representation which is manifested in the low values of bit scores are not considered for the final taxonomic assignment.

The sensitivity values obtained for the simulated prokaryotic metagenomic dataset in Table 3 are considerably high compared to those observed for the viral dataset. This is attributed to the fact that more prokaryotic genomes have been sequenced and are available in the reference database that was used for simulations. There were more distinct taxonomy ids for the prokaryotic database (5458) than for the viral database (2506), implying that the process of generation of test sequence data had more representation in the former. These distinct taxonomy ids represent organisms for which a substantial amount of sequence data is available as determined by the NCBI Taxonomy group. Also, the proportion of correctly classified sequence reads as seen from the sensitivity was higher for the weighted MRCA than for the unweighted MRCA. This demonstrates the ability and discriminatory power of the weighted algorithm compared to the original method that was used in the PPP. The accuracy values (Figure 10) obtained account for how well this assignment is close to the actual (true) value, and it can be seen from the table that 94.8% (at the rank of species) of the input dataset was correctly assigned by the weighted

MRCA, compared to 86.4% attained by the unweighted MRCA. At the $\alpha = 0.05$ level of significance, there is enough evidence to conclude (Table 3) that there is a difference in the classification performance of the two methods; MRCA and weighted MRCA. All the *p*-values obtained are below 0.05 (0.014-0.003), thus we reject the null hypothesis that the difference between the classification accuracy of MRCA and that of weighted MRCA is equal to zero.

The accuracy of assignments improves greatly as one moves higher up the taxonomy tree (i.e. from species to family). This is expected since some organisms may have only been characterised up to the taxonomic ranks of genus or family, but not at a species level.

A great number of metagenomic samples are derived from previously uncharacterized ("virgin") habitats, for which no prior sequence information is present in genome sequence repositories (Yooseph *et al.*, 2007). For this reason, a similarity search may not always generate the desired result but rather an approximation to the actual solution. The ability to design algorithms or methods that are able to discriminate against sequences and logically place them into taxonomic groupings is a desirable property and an important task for any metagenomic analysis (Huson *et al.*, 2009). A metagenome includes a massive diversity of microorganisms originating from a particular environment, with complex interactions between and within species. As such, information about the genetic variation of each of the species and its significance within the ecosystem is unknown (Wooley *et al.*, 2010).

In the context of our original goal (improving taxonomic assignment accuracy), what we ultimately want to do is to place the reads as accurately and reliably as possible in the taxonomic tree so the results are as informative as possible. Candidate pathogens can be identified and confirmed with experimental tests. Regardless of the method used, having assignments collapse back to the genus or family level is not desirable if a species level determination could have been possible. It was also observed that during pipeline runs, BLAST low-complexity filtering rendered the weighted MRCA algorithm less rigorous by drastically reducing the number of hits of a given input set against the same database. Since these short sequences had been simulated in a way that minimized complexity, this option was turned off in subsequent pipeline runs with the weighted MRCA.

The weighted MRCA approach implemented in the pipeline demonstrates the ability to improve this taxonomic mapping by just taking into consideration the bit score of a given read in the context of the overall dataset. In contrast to MEGAN and the previous MRCA approaches (Figure 10), the MRCA when calculated sometimes leads to a more general taxonomic assignment. This arises as a result of a few reads belonging to an outlier taxon (with high bit scores) biasing the assignment to a much higher level in the taxonomic tree.

Among the two approaches tested (MEGAN, MRCA), the weighted MRCA approach provides a better assignment at all taxonomic levels (Figure 9, 10). From our simulations, in the worst case, the algorithm would perform exactly as the original unweighted MRCA algorithm but never worse. Although not all assignments are resolved to the species and genus levels, the approach immensely reduces taxonomic mapping at "no rank" which was the biggest problem with the original MRCA method.

Metagenomic data with its diversity in terms of microbial origin normally consists of organisms from all domains of life. The sequence databases that are searched against are usually large and with more organisms being sequenced and deposited in these databases (Messing and Llaca, 1998), the computational time and cost of an analysis is overwhelming. It was thus imperative to assess the classification performance of the methods analyzed in this study on taxonomic accuracy, rather than computational time.

The QSA Read simulator is designed as a procedure that relies on a random number generator, which makes it almost impossible to generate the same metagenome sequence more than once. Additionally, all the sequencing errors (insertions, deletions, substitutions and inversions) that would normally be found in an experimental dataset were modelled into the application. Abundance files were also provided to guide the program on diversity of the different metagenomes expected. This was a key safeguard against biases, over fitting and uneven distribution of genome sequences or clustering in a single genome class.

CHAPTER FIVE: CONCLUSION

5.1 Research contributions

This work has shown that the weighted MRCA algorithm improves taxonomic assignment accuracy for unknown metagenomic reads. This demonstrates that rather than just mapping reads to their most recent common ancestor, summing up the bit scores and applying the weighting while maintaining mapping to a higher level, improves taxonomic assignments. On average, 92.8% specificity was obtained at species specific clustering for both prokaryotic and viral simulated data sets, compared to 89.2 % obtained using the other methods. These results showed an improved mapping of up to 3.6% for viral sequences and 8.4% for the prokaryotic sequences at the species level. MRCA calculations are greatly affected by assignments where for example one read is mapped in the NCBI taxonomy database with a high bit score yet this comprises a very small percentage in terms of the overall sample dataset. In all the simulated datasets, the weighted MRCA algorithm performed significantly better than MEGAN and the unweighted MRCA.

The weighted MRCA algorithm has been tested and implemented in the Pathogen Profiling Pipeline, a diagnostic tool that helps to inform researchers and the public community about possible causative pathogens during a disease outbreak. In the context of environmental science and medicine, the 3.6 and 8.4% improvement is highly significant as it informs key decisions in public health.

The application (weighted MRCA) was designed as a function that can be called from a module, thus facilitating portability and re-usability. All the Perl scripts can be customized to run on any Linux platform that meets the other dependences, like Bioperl, CPAN modules as would be required. Since the PPP software has been developed to be open source, this study has contributed to community software specific for bioinformatics. This research has also contributed to more accurate analysis and reporting of likely pathogenic microbes in a given metagenomic sample.

5.2 Limitations

Although the weighted MRCA achieved higher taxonomic classification accuracy compared to the original algorithm and MEGAN, other aspects like horizontal gene transfer (HGT) across species could not be assessed. Like with all other metagenomics analysis projects, the database limitation was a big challenge. This is because only those sequences for which close relatives exist in the database are likely to be mapped in the taxonomic tree (return BLAST hits). This in many cases implies that if a new organism is sequenced (as with all metagenomics projects) and no close relatives are present in the database, that sequence read will be returned as "Unassigned". This however, would not mean that the given sequence read does not have any taxonomic label, but that the taxonomic label is unknown to the method being used for taxonomic classification. Research is still ongoing to find ways of circumventing this database limitation (Huson *et al.*, 2009).

Simulations of metagenomic datasets are cheap, quick and easy. These provide a reliable estimate of what a pyrosequencing dataset would look like, if pyrosequencing errors and other factors that affect pyrosequencing data are taken into account during the simulations. Real laboratory metagenomic data, sampled from a known environment would provide a more realistic analysis of the taxonomic classification accuracy in that microbial population sample. This is because the environments from which these data are obtained tend to contribute to their diversity, a factor that is not accounted for in simulated datasets. However, the true taxonomic diversity present in a real sample is not known and therefore the accuracy of the taxonomic assignment approaches studied here cannot be addressed. The study was thus limited by the artificial nature of the simulated read sets; however, this was necessary to properly assess the taxonomic accuracy of the different taxonomic assignment approaches.

However, real laboratory metagenomic data, sampled from a known environment would provide a better analysis of the taxonomic diversity in that microbial population sample. This is because the environments from which these data are obtained tend to contribute to their diversity, a factor that is not accounted for in simulated datasets. The study was thus limited by the inability to make replicates in the simulations.

5.3 Recommendations for further research

The progress made during this study lends strong support for further analysis that includes closely examining each of the equivalent hits (for any read) in a given taxonomic sub tree (where the weighted MRCA maps). If the specific gene that the read mapped to could be captured, then a further analysis of this gene would reveal whether it is informative or not. Some genes are virtually everywhere, so such a revelation would be abandoned.

Since the data used was largely simulated, laboratory metagenomic datasets could be analyzed using the same algorithm in order to ascertain reproducibility, when a real (experimental] metagenome is used. This was because there were some outlier data samples that could not be considered for the final taxonomic assignments.

From the observed classification accuracies (~86-90%), this work lays a strong foundation for upscaling the accuracy of the clusters towards 100%. In future analyses, with experimental metagenomic datasets, this could be easily achieved.

CHAPTER SIX: BIBLIOGRAPHY

Abe T, Kanaya S, Kinouchi M, Ichiba Y, Kozuki T and Ikemura T (2002): A novel bioinformatic strategy for unveiling hidden genome signatures of eukaryotes: self-organizing map of oligonucleotide frequency. Genome Inform 13:12-20.

Altschul SF, Gish W, Miller W, Myers EW and Lipman DJ (1990): Basic local alignment search tool. J Mol Biol 215:403–410.

Amann R, Wolfgang L and Karl-Heinz S (1995): Phylogenetic identification and in situ detection of individual microbial cells without cultivation. Microbiol. Rev. 59:143-69.

Aragone`s L, Guix S, Ribes E, Bosch A and Pinto' RM (2010): Fine-Tuning Translation Kinetics Selection as the Driving Force of Codon Usage Bias in the Hepatitis A Virus Capsid. PLoS Pathog 6(3): e1000797. doi:10.1371/journal.ppat.1000797.

Ashelford KE, Chuzhanova NA, Fry JC, Jones AJ and Weightman AJ (2005): At Least 1 in 20 16S rRNA Sequence Records Currently Held in Public Repositories Is Estimated To Contain Substantial Anomalies. Applied and Environmental Microbiology, pp. 7724–7736 doi:10.1128/AEM.71.12.7724–7736.2005.

Benson DA, Karsch-Mizrachi I, Lipman DJ, Ostell J, Sayers EW (2011): GenBank. Nucleic Acids Res. 2011 Jan;39 (Database issue):D32-37.Epub 2010 Nov 10.

Biers EJ, Sun S and Howard EC (2009): Prokaryotic genomes and diversity in surface ocean waters: interrogating the global ocean sampling metagenome. Appl Environ Microbiol 75:2221–2229.

Brady A and Salzberg SL (2009): Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models. *Nature Methods*, 6:673-676.

Chatterji S, Yamazaki I, Bai Z, Eisen JAVingron M and Wong L (2008): CompostBin: A DNA composition-based algorithm for binning environmental shotgun reads. Lecture Notes in Computer Science Springer. 17-28.

Chan CK, Hsu AL, Tang S and Halgamuge SK (2008): Using Growing Self-Organising Maps to Improve the Binning Process in Environmental Whole-Genome Shotgun Sequencing.

Journal of Biomedicine and Biotechnology, Article ID 513701, doi:10.1155/2008/513701.

Chen K and Patcher L (2005): Bioinformatics for whole-genome shotgun sequencing of microbial communities. PLoS Comp Biol 1(2):e24.

Ciccarelli FD, Doerks T, von Mering C, Creevey CJ, Snel B and Bork P (2006): Toward automatic reconstruction of a highly resolved tree of life. Science, 311(57651283-1287).

Cole JR, Chai B, Farris RJ, Wang Q, Kulam SA, McGarrell DM, Garrity GM and Tiedje JM (2005): The Ribosomal Database Project (RDP-II): sequences and tools for high-throughput rRNA analysis. Nucleic Acids Res. 33, D294–D296.

Cox-Foster DL, Conlan S, Holmes CE, Palacios G, Evans JD, Moran AN, Quan PL, Briese T, Hornig M, Geiser MD, Martinson V, van Engelsdorp D, Kalkstein LA, Drysdale A, Hui J, Zhai J, Cui L, Hutchison KS, Simons JF, Egholm M, Pettis JS and Lipkin WI (2007): A metagenomic survey of microbes in honey bee colony collapse disorder. Science. 318(5848): p. 283-287.

Cuddapah S, Barski A, Cui K, Schones DE, Wang Z, Wei G, and Zhao K (2009): Native Chromatin Preparation and Illumina/Solexa Library Construction. Cold Spring Harb. Protoc.doi:10.1101/pdb.prot5237.

DeLong EF, Preston, CM, Mincer T, Rich V, Hallam SJ, Frigaard NU, Martinez A, Sullivan MB, Edwards R, Rodriguez Brito, B, Chisholm SW and Karl DM (2006): Community genomics among stratified microbial assemblages in the ocean's interior. Science 27: 496–503.

Diaz NN, Krause L, Goesmann A, Niehaus K and Nattkemper TW (2009): TACOA: taxonomic classification of environmental genomic fragments using a kernelized nearest neighbor approach. BMC Bioinformatics 10:56.

Ghazanfar S, Azim A, Ghazanfar MA, Anjum MI and Begum I (2010): Metagenomics and its application in soil microbial community studies: biotechnological prospects. Journal of Animal and Plant Sciences, 2010. Vol. 6, Issue 2: 611- 622.

Finn RD, Mistry J, Schuster-Bockler B, Griffiths-Jones S, Hollich V, Lassmann T, Moxon S, Marshall M, Khanna A, Durbin R, Eddy SR, Sonnhammer EL and Bateman A (2006): Pfam: clans, web tools and services. Nucleic Acids Res 34:D247–251.

Hallam SJ, Putnam N, Preston C, Detter J, and Rokhsar D (2004): Reverse methanogenesis: Testing the hypothesis with environmental genomics. Science 305: 1457–1462.

Handelsman J (2004): Metagenomics: application of genomics to uncultured microorganisms. Microbiol. Mol. Biol. Rev. 68, 669–685.

Harkins N, Kirby E, Heimsath A, Robinson R and Reiser U (2007): Transient fluvial incision in the headwaters of the Yellow River, northeastern Tibet, China, *J. Geophys. Res.*, 112, F03S04, doi:10.1029/2006JF000570.

Harris TD, Buzby PR, Babcock H, Beer E, Bowers J, Braslavsky I, Causey M, Colonell J, DiMeo J, Efcavitch JW, Giladi E, Gill J, Healy J, Jarosz M, Lapen D, Moulton K, Quake SR, Steinmann K, Thayer E, Tyurina A, Ward R, Weiss H and Xie Z (2008): Single-molecule DNA sequencing of a viral genome. *Science* 320, 106–109.

Hugenholtz P. (2002): Exploring prokaryotic diversity in the genomic era. Genome Biol. 3. REVIEWS0003.

Huse MS, Huber AJ, Morrison HG, Sogin ML and Welch MD (2007): Accuracy and quality of massively parallel DNA pyrosequencing .Genome Biology, 8:R143.

Huson DH, Auch AF, Qi J and Schuster SC (2007): MEGAN analysis of metagenomic data. Genome Res 17: 377-386.

Huson DH, Richter DC, Suparna M, Auch AF and Schuster SC (2009): Methods for comparative metagenomics. BMC Bioinformatics, 10(Suppl 1):S12.

Jolliffe IT (2002): Principal Component Analysis. Springer, Heidelberg.

Krause L, Diaz NN, Goesmann A, Kelley S, Nattkemper TW, Rohwer F, Edwards RA and Stoye J (2008): Phylogenetic classification of short environmental DNA fragments. Nucleic Acids Res 36:2230–2239.

Kunin V, Copeland A, Lapidus A, Mavromatis K and Hugenholtz P (2008): A Bioinformatician's Guide to Metagenomics. Microbiol. Mol. Biol. Rev. 4, 557-578. doi:10.1128/MMBR.00009-08.

Larkin MA, Blackshields G, Brown NP, Chenna R, McGettigan PA, McWilliam H, Valentin F, Wallace IM, Wilm A, Lopez R, Thompson JD, Gibson TJ and Higgins DG (2007): Clustal W and Clustal X version 2.0. *Bioinformatics (Oxford, England). 23 (21):2947-8.*

Lozupone CA and Knight R (2007): Global patterns in bacterial diversity. Proc Natl Acad Sci U SA 104, 11436-11440.

Manichanh C, Chapple CE, Frangeul L, Gloux K, Guigo R and Dore J (2008): A comparison of random sequence reads versus 16S rDNA sequences for estimating the biodiversity of a metagenomic library. Nucleic Acids Res 36:5180–5188.

Margulies M, Egholm M, Altman WE, Attiya S, Bader JS, Bemben LA, Berka J, Braverman MS, Chen Y, Chen Z, Dewell SB, Du L, Fierro JM, Gomes XV, Godwin BC, He W, Helgesen S, Ho CH, Irzyk GP, Jando SC, Alenquer MLI, Jarvie TP, Jirage KB, Kim JB, Knight JR, Lanza JR, Leamon JH, Lefkowitz SM, Lei M, Li J, Lohman KL, Lu H, Makhijani VB, McDade KE, McKenna MP, Myers EW, Nickerson E, Nobile JR, Plant R, Puc BP, Ronan MT, Roth GT, Sarkis GJ, Simons JF, Simpson JW, Srinivasan M, Tartaro KR, Tomasz A, Vogt KA, Volkmer GA, Wang SH, Wang Y, Weiner MP, Yu P, Begley RF and Rothberg JM (2005): Genome sequencing in microfabricated high-density picolitre reactors. Nature 437: 376-380.

Martin C, Diaz NN, Ontrup J and Nattkemper TW (2008): Hyperbolic SOM-based clustering of DNA fragment features for taxonomic visualization and classification. Bioinformatics 24, Issue 14:1568-1574.

Martin HG, Ivanova N, Kunin V, Warnecke F, Barry K, McHardy AC, Yeates C, He S, Salamov A, Szeto E, Dalin E, Putnam N, Shapiro HJ, Pangilinan JL, Rigoutsos I, Kyrpides NC, Blackall LL, McMahon KD and Hugenholtz P (2006): Metagenomic analysis of phosphorus removing sludge communities. Lawrence Berkeley National Laboratory: Lawrence Berkeley National Laboratory. LBNL Paper LBNL-59661. Retrieved from: http://escholarship.org/uc/item/30z138fd.

Matthews T, Kent H, Tyler S, Bonner C, Peters G, Bristow F, Mabon P, Graham M, and Van Domselaar G (2009): Pathogen Profiling Pipeline: A metagenomics tool for rapid identification of pathogens from clinical specimens. Unpublished data.

McHardy AC, Martin HG, Tsirigos A, Hugenholtz P and Rigoutsos I (2007): Accurate phylogenetic classification of variable-length DNA fragments. Nat Methods 4:63–72.

Messing J and Llaca V (1998): Importance of anchor genomes for any plant genome project. PNAS March 3, vol. 95 no. 5 2017-2020.

Monzoorul HM, Tarini S, Dinakar K and Sharmila SM (2009): SOrt-ITEMS: sequence orthology based approach for improved taxonomic estimation of metagenomic sequences. Bioinformatics 25:1722–1730.

Morgan JL, Darling AE and Eisen JA (2010): Metagenomic Sequencing of an *In Vitro*-Simulated Microbial Community. PLoS ONE 5(4): e10209. doi:10.1371/journal.pone.0010209.

Nakamura S, Yang CS, Sakon N, Ueda M, Tougan T, Yamashita A, Goto N, Takahashi K, Yasunaga T, Ikuta K, Mizutani T, Okamoto Y, Tagami M, Morita R, Maeda N, Kawai J, Hayashizaki Y, Nagai Y, Horii T, Iida T, Nakaya T (2009): Direct Metagenomic Detection of Viral Pathogens in Nasal and Fecal Specimens Using an Unbiased High-Throughput Sequencing Approach. PLoS ONE. 4(1):e4219.

Ontrup J and Ritter H (2006): Large scale data exploration with the hierarchical growing hyperbolic SOM. Neural Netw 19:751-761.

Pace NR (1997): A Molecular View of Microbial Diversity and the Biosphere. *Science* 276, 734. doi: 10.1126/science.276.5313.734.

Pachter L (2007): Interpreting the unculturable majority. Nat Methods 4: 479-480.

Palacios G, Druce J, Du L, Tran T, Birch C, Briese T, Conlan S, Quan PL, Hui J, Marshall J, Simons JF, Egholm M, Paddock CD, Shieh WJ, Goldsmith CS, Zaki SR, Catton M and Lipkin WI (2008): A New Arenavirus in a Cluster of Fatal Transplant-Associated Diseases. New England Journal of Medicine, 358: p. 991-998.

Patrick J. Deschavanne PJ, Giron A, Vilain J, Fagot G, and Fertil B (1999): Genomic Signature: Characterization and Classification of Species Assessed by Chaos Game Representation of Sequences. Mol. Biol. Evol. 16(10):1391–1399.

Pertsemlidis A and Fondon III JW (2001): Having a BLAST with bioinformatics (and avoiding BLASTphemy). *Genome Biology*, 2(10):reviews2002.1–2002.10.

Petrosino JF, Highlander S, Luna RA, Gibbs RA and Versalovic J (2009): Metagenomic pyrosequencing and microbial identification. *Clin. Chem.* 55, 856–866.

Poinar HN, Schwarz C, Qi J, Shapiro B, Macphee RD, Buigues B, Tikhonov A, Huson DH, Tomsho LP, Auch A, Rampp M, Miller W and Schuster SC (2006): Metagenomics to paleogenomics: large-scale sequencing of mammoth DNA. Science 311:392–394.

Pruitt KD, Tatusova T, Maglott DR (2005): NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. Nucleic Acids Res. 2005 Jan 1;33 (Database issue):D501-4.

Quince C, Lanzen A, Curtis TP, Davenport RJ, Hall N, Head IM, Read LF and Sloan WT (2009): Accurate determination of microbial diversity from 454 pyrosequencing data. Nat. Methods 6, 639. doi:10.1038/NMETH.1361.

Raes J, Foerstner UK and Bork P (2007): Get the most out of your metagenome: computational analysis of environmental sequence data. Current Opinion in Microbiology, 10:490–498.

Riesenfeld CS, Schloss PD and Handelsman J (2004): METAGENOMICS: Genomic Analysis of Microbial Communities Annu. Rev. Genet. 2004. 38:525–52 doi: 10.1146/annurev.genet.38.072902.091216.

Rodriguez-Brito B, Rohwer F and Edwards RA (2006): An application of statistics to comparative metagenomics. BMC Bioinformatics 7: 162.

Rusch, DB, Halpern AL, Sutton G, Heidelberg KB, Williamson S, Yooseph S, Wu D, Eisen JA, Hoffman JM, Remington K, Beeson K, Tran B, Smith H, Baden-Tillson H, Stewart C, Thorpe J, Freeman J, Andrews-Pfannkoch C, Venter JE, Li K, Kravitz S, Heidelberg JF, Utterback T, Rogers YH, Falcon LI, Souza V, Bonilla-Rosso, G, Eguiarte LE, Karl DM, Sathyen-dranath S, Platt T, Bermingham E, Gallardo V, Tamayo-Castillo G, Ferrari MR, Strausberg RL, Nealson K, Friedman R, Frazier M and Venter JC. (2007): The Sorcerer II global ocean sampling expedition: Northwest atlantic through eastern tropical pacific. PLoS Biol. 5(3), e77.

Richter DC, Ott F, Auch AF, Schmid R and Huson DH (2008): MetaSim—A Sequencing Simulator for Genomics and Metagenomics. PLoS ONE 3(10): e3373. doi:10.1371/journal.pone.0003373.

Rolf D and Carola S (2009): Achievements and new knowledge unraveled by metagenomic approaches. Appl Microbiol Biotechnol. November; 85(2): 265–276. doi: 10.1007/s00253-009-2233-z.

Salzberg SL, Delcher AL, Kasif S, and White O (1998): Microbial gene identification using interpolated Markov models. *Nucleic Acids Research*, 26:2, 544-548.

Sanger F, Air GM, Barrell BG, Brown NL, Coulson AR, Fiddes JC, Hutchinson CA, Slocombe PM and Smith M (1977): The nucleotide sequence of bacteriophage phi X174 DNA. Nature 265:687-695.

Sayers EW, Barrett T, Benson DA, Bryant SH, Canese K, Chetvernin V, Church DM, DiCuccio M, Edgar R, Federhen S, Feolo M, Geer LY, Helmberg W, Kapustin Y, Landsman D, Lipman DJ, Madden TL, Maglott DR, Miller V, Mizrachi I, Ostell J, Pruitt KD, Schuler GD, Sequeira E, Sherry ST, Shumway M, Sirotkin K, Souvorov A, Starchenko G, Tatusova TA, Wagner L, Yaschenko E, Ye J (2009): Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 2009 Jan;37 (Database issue):D5-15.

Schbath S, Prum B and de Turckheim E (1995): Exceptional motifs in different Markov chain models for a statistical analysis of DNA sequences.J Comput Biol, 2:417-437.

Schbath S (1997): An efficient statistic to detect over- and under-represented words in DNA sequences. J Comput Biol, 4:189-192.

Schloss PD and Handelsman J (2005): Introducing DOTUR, a Computer Program for Defining Operational Taxonomic Units and Estimating Species Richness. Appl. Environ. Microbiol. 71, 1501–1506.

Schmieder R and Edwards R (2011): Fast Identification and Removal of Sequence Contamination from Genomic and Metagenomic Datasets. PLoS ONE 6(3): e17288. doi:10.1371/journal.pone.0017288.

Schuster SC (2008): Next-generation sequencing transforms today's Biology. Nature methods 5(1) DOI:10.1038/NMETH1156.

Shah N, Tang H, Doak TG and Ye Y (2010): Comparing bacterial communities inferred from 16s rRNA gene sequencing and shotgun metagenomics. In WSPC – Proceedings September 20, 2010.

Sogin ML, Morrison HG, Huber JA, Welch DM, Huse SM, Neal PR, Arrieta JM, and Herndl GJ (2006): Microbial diversity in the deep sea and the underexplored ''rare biosphere''. PNAS 103:12115-12120 doi_10.1073_pnas.0605127103.

Staley JT and Konopka A (1985): Measurement of in Situ Activities of Non-photosynthetic Microorganisms in Aquatic and Terrestrial Habitats. Annual Review of Microbiology 3 9:321-46, doi:10.1146/annurev.mi.39.100185.001541.

Steele LH and Streit WR (2005): Metagenomics: Advances in ecology and biotechnology FEMS Microbiology Letters 247 105–111 MiniReview.

Tatusov RL, Wolf YI, Rogozin IB, Grishin NV and Koonin EV (2001): Genome trees constructed using five different approaches suggest new major bacterial clades. BMC Evol. Biol. 1, 8.

Teeling H, Waldmann J, Lombardot T, Bauer M and Glöckner FO (2004): TETRA: a web-service and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in DNA sequences. BMC Bioinformatics;5(163).

Tringe SG, von Mering C, Kobayashi A, Salamov AA, Chen K, Chang HW, Podar M, Short JM, Mathur EJ, Detter JC, Bork P Hugenholtz P and Rubin EM (2005): Comparative Metagenomics of Microbial Communities. Science 308: 554-557.

Turnbaugh PJ, Ley RE, Hamady M, Fraser-Liggett CM, Knight R and Gordon JI. (2007). The human microbiome project. Nature 449, 804-810.

Tyson GW, Chapman J, Hugenholtz P, Allen EE, Ram RJ, Richardson PM, Solovyev VV, Rubi EM, Rokhsar DS and Banfield JF (2004): Community structure and metabolism through reconstruction of microbial genomes from the environment. Nature 428: 37-43.

Valouev A, Ichikawa J, Tonthat T, Stuart J, Ranade S, Peckham H, Zeng K, Malek JA, Costa G, McKernan k, Sidow A, Fire A, and Johnson SM (2008): A high-resolution, nucleosome position map of *C. elegans* reveals a lack of universal sequence-dictated positioning. *Genome Res.* 18, 1051–1063.

Venter JC, Remington K, Heidelberg JF, Halpern AL, Rusch D Eisen JA, Wu D, Paulsen I, Nelson KE, Nelson W, Foutus DE, Levy S, Knap AH, Lomas MW, Nealson K, White O, Peterson J, Parsons R, Baden-Tillson H, Pfannkock C, Rogers YH and Smith HO (2004): Environmental Genome Shotgun Sequencing of the Sargasso Sea. Science 304: 66-74.

Woese CR and Fox GE. (1977): Phylogenetic structure of the prokaryotic domain: the primary kingdoms. Proc. Natl. Acad. Sci. USA 74, 5088-5090.

Wooley JC, Godzik A and Friedberg I (2010): A Primer on Metagenomics. PLoS Comput Biol 6(2): e1000667. doi:10.1371/journal.pcbi.1000667.

Wu D, Hartman A, Ward N and Eisen JA (2008): An Automated Phylogenetic Tree-Based Small Subunit rRNA Taxonomy and Alignment Pipeline (STAP). PLoS ONE 3(7): e2566. doi:10.1371/journal.pone.0002566

Yooseph S, Sutton G, Rusch DB, Halpern AL, Williamson SJ, Remington K, Eisen JA, Heidelberg KB, Manning G, Li W, Jaroszewski L, Cieplak P, Miller CS, Li H, Mashiyama ST, Joachimiak MP, van Belle C, Chandonia JM, Soergel DA, Zhai Y, Natarajan K, Lee S, Raphael BJ, Bafna V, Friedman R, Brenner SE, Godzik A, Eisenberg D, Dixon JE, Taylor SS, Strausberg RL, Frazier M and Venter JC (2007) The Sorcerer II global ocean sampling expedition: expanding the universe of protein families. PLoS Biol 5:e16.

Zhou J (2009): Predictive microbial ecology. Microbial Biotechnology, 2: 154–156. doi: 10.1111/j.1751-7915.2009.00090_21.x.

CHAPTER SEVEN: APPENDICES

7.1 Appendix A

i) Installing and setting up the Pathogen Profiling Pipeline (PPP)

PPP was downloaded from: http://www.corefacility.ca/ppp

The download contained 2 program directories:

- ppp-backend - Filtering and execution management software
- ppp-web - The web front

From the README in ppp.tar.gz the following software requirements had to be met namely;

```
Compute cluster:
     - BLAST
     - BioPerl -- 1.5 or newer
     - DRMAA compliant scheduler -- Sun Grid Engine suggested
 Web server:
     - Apache2
     - Mod-Perl
     - BioPerl -- 1.5 or newer
     - Graphviz
```

BLAST installation
The latest BLAST version was downloaded, installed and configured thus

```
$ cd $HOME/Downloads
wget ftp://ftp.ncbi.nih.gov/blast/executables/release/LATEST/blast-
2.2.23-x64-linux.tar.gz # as of May 05, 2010.
sudo tar xfv blast-2.2.23-x64-linux.tar.gz -C /usr/local && \
sudo ln -s /usr/local/blast-2.2.23 /usr/local/blast
```

The BLASTMAT and BLASTDIR environment variables were added to the `/etc/bash.bashrc` file

```
$ sudoedit /etc/bash.bashrc
```

The bashrc was modified like this:

```
PATH=/usr/local/blast/bin:$PATH
BLASTMAT=/usr/local/blast/data
BLASTDIR=/usr/local/blast/bin
export BLASTMAT BLASTDIR PATH
```

Now source the bashrc file and test that blast is in the path:

```
$ source /etc/bash.bashrc && which blastall
```

```
        /usr/local/blast/bin/blastall
```

Installing BioPerl

BioPerl heavily depends on the Perl programming language itself. Ubuntu (v10.04) already comes with the latest version of Perl (5.10.1). The latest version of BioPerl was downloaded from http://www.bioperl.org/wiki/Installing_Bioperl_for_Unix.

This was unpacked like so;

```
        $ tar xvfz BioPerl-1.6.1.tar.gz
        $ cd BioPerl-1.6.0
```

And built using these commands

```
        $ perl Build.PL
        $ sudo ./Build test
        $ sudo ./Build install
```

Installation and Configuration of the head node ---Web server

The Apache2 web server was downloaded and installed from http://httpd.apache.org/. Mod-perl was installed from the Yum Package Manager. Yum is an automatic updater and package installer/remover for RPM systems. It automatically computes dependencies and figures out what things should occur to install packages. It makes it easier to maintain groups of machines without having to manually update each one using rpm.

Graphviz was downloaded from http://www.graphviz.org/Download_linux_rhel.php and copied to `/etc/yum.repos.d/`.

The install was done thus

```
        $ sudo yum install 'graphviz*'
```

Configuring Perl modules

Since PPP's web interface requires XML::Simple, the module was installed from yum:

```
        $ sudo yum install perl-XML-Simple
```

PPP's DRMAA scheduler

The PPP interacts with the scheduler software thus the need for a binding to the DRMAA.

DRMAA is an Application Programming Interface (API) for job scheduling. Sun Grid Engine is DRMAA compliant, but it needs the help of a Perl module. DRMAA was compiled from source because we need to tell it where to look to find the C headers for SGE's drmaa support.

Schedule/DRMAAc was downloaded from CPAN: http://search.cpan.org/CPAN/authors/id/T/TH/THARSCH/Schedule-DRMAAc-0.81.tar.gz. From the README, the environment for compiling the Perl module was prepared like so:

```
$ source /opt/gridengine/default/common/settings.sh
$ export LD_LIBRARY_PATH=$SGE_ROOT/lib/`$SGE_ROOT/util/arch`
$ ln -s $SGE_ROOT/include/drmaa.h
```

Build and install the Perl module:

```
$ perl Makefile.PL
$ make
$ make test
$ sudo make install
```

PPP Backend

An appropriate directory was chosen for the PPP management software to run and the contents of the ppp-backend folder copied to this location. The binary directories were placed in a shared location such that they could be accessed by all nodes in the cluster. The location chosen was the /opt directory.

Install PPP

PPP's Perl scripts need to be accessible to all nodes, so the directory was changed to somewhere accessible:

```
$ cd /opt
$ sudo mkdir apps
$ cd apps
```

Unzip PPP:

```
$ tar -zxf ppp.tar.gz
```

From the README, the install instructions in INSTALL.PDF were followed thus … In a nutshell:

```
$ cd ppp-backend
```

Data Directories

The following directories required by the PPP were created inside the ppp-backend directory.

- data - the directory where input data files are stored
- db - the base directory to the database repository
- scratch - the working and storage directory for all running and completed executions
- taxon - a location for the NCBI taxonomy databases

```
  $ mkdir db scratch data
```

Taxonomy databases

In order to use some advanced features that the PPP offers, such as the most recent common ancestor calculations, the latest taxonomy databases were downloaded from the NCBI's FTP site, ftp://ftp.ncbi.nih.gov/pub/taxonomy/taxdump.tar.gz

```
$ wget ftp://ftp.ncbi.nih.gov/pub/taxonomy/taxdump.tar.gz

$ cd taxon
$ tar zxf taxdump.tar.gz
```

From the taxon folder, the taxonomy databases were formatted using the `taxonformat.pl` script from Tom Matthews (PPP developer):

```
#!/usr/bin/perl

use Bio::DB::Taxonomy;
use Bio::Taxonomy::Taxon;
use FindBin;
use strict;

my $taxondir = $FindBin::Bin;
if(!-d $taxondir)
{
        die "$taxondir is not a directory";
}
if(!-e "$taxondir/nodes")
{
        print  "It  doesn't  look  like  $taxondir  is  formatted.   Formatting
$taxondir\n";
}
my $db = new Bio::DB::Taxonomy(-source => 'flatfile',
        -directory => $taxondir,
        -nodesfile => "$taxondir/nodes.dmp",
        -namesfile => "$taxondir/names.dmp");

if(-e "$taxondir/nodes" && -e "$taxondir/id2names" && -e "$taxondir/names2id"
&& -e "$taxondir/parents")
{
        print "Success!  Taxonomy directory $taxondir appers to be properly
formatted.\n";
}
else
{
        print "There may be an error.  Check to ensure that $taxondir has the
extracted taxonomy database and try again\n";
}
```

PPP Configuration

The configuration file for the pipeline, *ppp-backend/conf/local.conf* was edited to reflect the locations of software in the installation.

- blast_loc - the path to the blastall executable
- formatdb - the path to the formatdb executable
- bp_index_loc - the path to bp_index.pl
- root path - the full path to the ppp-backend/bin directory
- num_segments - the number of segments the clustered execution will be split into.

This was set around 2-3 times the number of CPU cores in the cluster. The larger this value is set, the more evenly work will be distributed over the cluster nodes, but this also increases the overhead time of waiting on the scheduler.

Static Job Setup

This was set up in the ppp-backend/bin/ directory and the script customjob.pl run with no arguments. The output of this file was placed in the ppp-backend/conf/ directory in a file jobs.xml.

```
$ perl bin/customjob.pl > conf/jobs.xml
```

PPP Web Front

The PPP is designed such that the analysis steps are done only on the back-end, so the web server should not need to do any particularly heavy work.

PPP Software

The software was installed in Apache's web root under its own directory i.e, /var/www/ppp-web/ and the permissions configured. A connection was created to the PPP back-end folder by making a symbolic link.

```
    $ sudo cp -R ppp-web/ /var/www/ppp-web/
    $ sudo ln -s /opt/apps/ppp-backend  /var/www/ppp-web/ppp
```

Apache configuration

Apache's config file was edited to load PPP as Perl scripts. A new file `/etc/httpd/conf.d/ppp-web.conf` was created:

```
<IfModule mod_perl.c>
        <Directory "/var/www/html/ppp-web">
                AllowOverride None
                Order allow,deny
                allow from all
                AddHandler perl-script cgi-script .cgi .pl
                Options None
        </Directory>

        <Directory "/var/www/html/ppp-web/cgi-bin">
                AllowOverride None
                Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
                Order allow,deny
                Allow from all
                SetHandler perl-script
                PerlResponseHandler ModPerl::Registry
        </Directory>
</IfModule>
```

Restart Apache:

```
    $sudo apachectl restart
```

The permissions on everything were changed so that Apache's user can read/write:

```
    $ sudo chown -R root:apache *
    $ sudo chmod -R g+w *
```

Test! http://hpc.ilri.cgiar.org/ppp-web/cgi-bin/pppweb.pl

If there are any errors, check the Apache error_log.

If it worked, the job manager is started thus (from the ppp-backend/bin directory):

```
$ cd bin
$ perl drmaamanager.pl -v
> Job manager initilized...
```

Now PPP's web interface should indicate that there is a job server running (green circle!)

Porting data into the PPP

For testing, sequence data was obtained from ftp://ftp.ncbi.nih.gov/refseq/release/microbial/ for the microbial NCBI FTP sequences site and ftp://ftp.ncbi.nih.gov/refseq/release/viral/ for viral sequence data, uploaded into PPP using the web interface and formatted for BLAST using *formatdb* utility.

Creating BLASTable databases

The database files after being downloaded were concatenated thus;

```
$ cat *.fna > microbial.fna (for refmicrobial database)
```

The concatenated database files were blast-formatted using the command

```
$ formatdb -i microbial.fna -p F
```

A Bioperl index was created by selecting the "Bioperl index" button this option was already implemented in the PPP

Managing the pipeline

Manually adding databases

- Add the FASTA files to your 'db' directory.
- Format the database with the formatdb utility included with BLAST. The command "formatdb --help" provides you with the appropriate arguments.
- If you would like a BioPerl index, you can also make it manually. Running "bp_index.pl" with no arguments will provide you with a perldoc page for the script,

```
$ bp_index.pl -dir <full path to database directory>
microbial.fna.idx microbial.fna
```

Adding Input Files

From the Administration page, click the Upload Files button. From here adding input files is very similar to adding databases.

Again note that you can manually add the files to your data directory from the command line if you wish. Seeing they don't need to be formatted, they will be ready to use as soon as they are placed in the appropriate directory.

Troubleshooting

CHECK THIS FIRST - If a problem occurred with the entry point script it may have locked the job cache. Ensure "ppp.pl" is not running on the web server or head cluster node, then remove the "ppp-backend/cache/jobcache.lock" file if it exists. This may resolve all kinds of problems.

Job manager error message: *Could not contact DRM system* - Your scheduler is not started. If using SGE, you need to start "sge_execd" on all execution hosts and "sge_qmaster" and "sge_schedd" on your submit host (head node).

Web front not displaying or trying to download pages - The apache2 configuration isn't properly set up. Check that the "ppp-web" apache2 configuration file is in apache2's "sites-available" folder and linked in "sites-enabled". Also ensure ModPerl (libapache2-mod-perl2) is installed. Finally, restart apache2 (apache2ctl restart).

Submitted jobs are not picked up by job manager - Check first that the job cache files are being created in "ppp-backend/cache". They will have the form "##.exec". If the files don't exist, it is probably a permissions problem. Ensure the apache2 web user has read/write access to the "ppp-backend/bin" and "ppp-backend/cache" folders.

Filtering jobs not producing results or immediately failing - Your paths may be set up wrong in the local configuration file. Look at "ppp-backend/conf/local.conf" and ensure all paths are set up correctly. Also, Sun Grid Engine may be failing. Check your gridengine/default/qmaster/messages file for a diagnosis of failing jobs.

Jobs appear to be running but producing no results - Again probably a permissions problem. Your web server is writing the job information, but the user running the execution jobs may not have read/write permissions to the scratch folders.

ii) Installation of MEGAN

MEGAN is written in Java and requires a Java runtime environment (JRE). The latest JRE was installed thus alongside its dependences;

```
$ sudo apt-get install sun-java6-fonts sun-java6-jdk sun-java6-jre sun-
java6-plugin sun-java6-source
```

Navigate to the directory where MEGAN was downloaded to;

```
$ cd ~/apps
```

The MEGAN installer shell script was executed as follows;

```
$ sh MEGAN_unix_3_9.sh
```

testing JVM in /usr ...

Starting Installer ...

After installation was completed, navigate to the directory in question

```
$ cd megan
```

The application was initiated as follows;

```
$ ./MEGAN
```

iii) The QSA Read simulator

The application was designed to be run on a modern Linux platform and required the following packages:

- Perl
- BioPerl, both of which were already installed.

The directory containing the application was obtained and placed in the applications folder, on a location on the computer from which the application was run. The QSA Read Simulator was initiated as follows and below shows the usage statement that details the options available.

```
$ cd ~/apps/QSAreadsim
$ ./readsim.pl
```

QSA Read Simulator - readsim.pl

Generates a simulated pyrosequencing dataset from one or more genomic FASTA sequences

Arguments:

-r | --reads: FASTA file of input reads to sample

-l | --lengths: Average length of generated reads

OR file containing input read lengths

-d | --stdev: Standard deviation of generated read length

-c | --coverage: Coverage depth to emulate over all input reads

-a | --abundfile: File of sequence abundances. Abundance format described above

-i | --id-prefix: Output read name prefix. Valid characters: letters, numbers, dashes, underscores

-n | --numreads: Number of reads to generate (This option will take priority over coverage)

-o | --output: Output fasta file

-p | --processes: Number of processes to run (default 1)

--no-errors: Do not generate any errors

--no-mates: Do not generate any mate-pair information

7.2 Appendix B: Simulated test data

Sample simulated data were generated by running this command. For example to generate 10000 sequence reads of average length 206 within a standard deviation of 10 and output them into a file, rather than displaying in the command prompt window.

```
$ ./readsim.pl -r /home/hellen/apps/data/refmicrobial.fna -l 206 -d 10
-i RM -n 10000 -o simrefmicrobial_microbial_reads.fna
```

Sample output data generated from running this command

```
$ less ~/apps/data/simrefmicrobial_microbial_reads.fna
```

```
>RM00001 NC_004310 location:234347-234153
GAAGGGCGCTGCCGGTGCTCGATGAANACGGCCGATCGCTTTGGCCGCGTGACATTGGAA
AAGCTACTGCATCAGGCAGAGCGCCCGGCATGAAGCTTTTCCTTTCCCTGTTTCCGCGCA
TGGTGCTTGTGGCGGTGCTGCTGCTCTTTCTCCTGCACCCGCATCTTTTCGAGCCGGTTT
TCCGCCCGTTCGTCN
>RM00002 NC_011333 location:507813-508017
ATTATGAAACACAACTTCTTACTTATAAATGGTGTTTTCTGCGATAGGTACGGAAAACCT
TTAAAAGGTGCGGCTCTGTATAATGCTAAAGCAGCATATTTGAATGGCGGTCTTAAGCAG
ACCTTAGATCAGATTGAAAGACTTAAAGATGAAAACAAGGGACTTAATGAAGCACTTTAT
TATTCCACTTTCTCTAACTCGAAC
>RM00003 NC_013342 location:10881-11087
TGCTCCCGTTTTGTNTCAAATTTGGAAGAAAAAACATAAAAAAGCGTATTATAAGTGGCG
TGTTGATGAGACATATATCAAAATTAAAGGACAGTGGTGTTATCTGTATCGCGCGATTGA
TGCAGATGGACATACATTAGATATTTGGTTGCGTAAGCAACGAGATAATCATTCAGCATA
TGCGTTTATCAAACGTCTCATTAAACA
>RM00004 NC_001849 location:11033-11224
TGCTATTTGATTTAGTAGCCTGTGTTGTGATTGATCTTTCAATTTTATTGATGGCTTGTA
```

AAAGCCTTATTTGGCCGTTTATTAAGGTTTTTTTTTCTATGTTTGAGTTTTAAAGGATTT

AATTTTTGATTCATTTTTGAATTTCTTTAATTTTCTTCTGCAGTTGGTTCATAATTTATC

CNTTTTTTAATCT

>RM00005 NC_005324 location:1321-1534

GCCGTCAATCCCGACTTCTTACCGGACGAGGACAAGAGTACGCCGCAGCTCGATCTTTTG

GCTCGTGTCGAACGAGAGCTACCGGTGCGGCTCGACCAAGAGCGCACCGATATGGTTGTT

TGCCATGGTGATCCCTGCATGCCGAACTTCATGGTGGACCCTAAAACTCTTCAATGCACG

GGTCTGATCGACCTTGGGCGGCTTGGAACAGGCAN

>RM00006 NC_013653 location:18689-18890

ACTTATTTCGTCCCCTACCTCATAGGATTCTTGATATAAATGTTTTAAATCATTGTTATC

ACTATAATCAAAGTCATATTCACTCAATAATTTCTTTTTGAATAGCCCCAAGTACAAATT

TATCATGCTGATTTTTATTAGGTTTAAATTCTTTTTCTTGTAACAACTTAACTTGTTCAG

TATATATTTTCTATCTTCACA

>RM00007 NC_013334 location:3979-4206

CCTTAAATTACTCTTTGANGCCAGCGACTAATNATAGACAAGCATTAATCCGCAAGAAGC

AACCTTTTACTGAAGAATATCAAAAAGCAACCAACAACAAAAGAAAATACAACCAATAGA

AGCAAATTAAATCATAAGGGCNTAAACTAAAGCGGAAAAAGGANGACAGTGCAAACGAAG

ACGTATTAAAGAGAGATTGATAAAACTANCTGAAAAAAACGAGNGTGGACCCT

....

7.3 Appendix B: Scripts

i) dl.pl

```
#!/usr/bin/perl
#dl.pl downloads a given database unattended from refseq
use strict;

for(my $i=1;$i<=65;$i++)
{
      my $cmd = "wget
ftp://ftp.ncbi.nih.gov/refseq/release/microbial/microbial$i.genomic.gbff.gz";
      print "$cmd\n";
      system($cmd);
      $cmd = "gunzip microbial$i.genomic.gbff.gz";
      system($cmd);
      $cmd = "perl /home/hellen/apps/scripts/convert.pl -i
microbial$i.genomic.gbff -o microbial$i.fna -t fasta";
      system($cmd);
}
```

ii) convert.pl

```
#!/usr/bin/perl
#convert.pl converts a genbank file into fasta format
use Bio::Seq;
use Bio::SeqIO;
use Getopt::Long;
use strict;

my ($fname,$outtype,$outfile);
Getopt::Long::Configure ('bundling');
GetOptions (
    "i|input=s"=>\$fname,
    "t|type=s"=>\$outtype,
    "o|fileout=s"=>\$outfile
);
if(!$fname || !$outtype || !$outfile)
{
    print "require -i -t -o\n";
    exit;
}

my $input = Bio::SeqIO->new(-file=>$fname,-format=>"genbank");
my $output = Bio::SeqIO->new(-file=>">$outfile",-format=>$outtype);
my $count = 0;
my $time = time;

while(my $seq = $input->next_seq)
{
    if($outtype eq 'fasta')
    {
        my $descline = $seq->desc." [".$seq->species->node_name."]";
        $seq->desc($descline);
    }
    $output->write_seq($seq);



    $count++;
    if($count%1000 == 0)
```

```
      {
            $time = time - $time;
            my $sps = 1000/$time;
            print "Analyzed $count entries - $sps/second\n";
            $time = time;
      }
}
```

iii) ranks_filter.pl

```
#!/usr/bin/perl
#ranks_filter.pl takes a ppp_output hits file,compares it with the original
#input file to get the accession numbers.The script then searches the index
#database file for the matching accession nos and returns the corresponding
#rank

use Getopt::Long;
use Bio::SeqIO;
use Bio::Index::Fasta;
use Bio::DB::Taxonomy;
use Bio::Taxonomy::Taxon;
use strict;
use warnings;

my ( $infile, $pppinput, $indexfile, $help );
GetOptions(
    'i|infile=s'          => \$infile,
    'p|ppp_input_file=s'  => \$pppinput,
    'd|indexfile=s'       => \$indexfile,
    'h|help'              => \$help
);
check_inputs( $infile, $pppinput, $indexfile, $help );
open(INFILE, "$infile");
my %names;
while (<INFILE>)
{
      chomp;
      if(/^\>(\S+) (.*) rank=\[[^\]]*\](.*)$/)
      {
            my $name = $1;
            my $desc = $3;
            $desc =~ s/^\s//g;
            $names{$name} = $desc;
      }
}
my $test;#just a flag
my %id_list = match_ids();
my $index = Bio::Index::Fasta->new(-filename => $indexfile);
my $taxondir = "/opt/apps/ppp-backend/taxon";

#create taxonomy outside the foreach loop,so that it doesnt do this every
time
my $db = new Bio::DB::Taxonomy(
    -source    => 'flatfile',
    -directory => "$taxondir",
    -nodesfile => "$taxondir/nodes.dmp",
    -namesfile => "$taxondir/names.dmp"
);

my $corr = 0;
my %qhash;
my %rankhash;
my $cur;
foreach my $read_id (keys(%id_list))
{
      my $seq_id   = $id_list{$read_id};
      my $descline = $index->fetch($seq_id);
      if ($descline->desc() =~ /([^\[]*)\]$/)  #get description starting from
the right(only what's in the []braces)
      {
            my $dbdesc = $1;
            if(($names{$read_id}) eq $dbdesc)
            {
                  $corr++;
                  my $qdesc = $names{$read_id};
                  my $taxid = $db->get_taxonid($qdesc);
                  my $taxon = $db->get_Taxonomy_Node($taxid);
                  my $found =0;
                  if($taxon->rank eq 'no rank')
                  {
                        my $tmp = $taxon;
                        while(defined($tmp) && $tmp->ncbi_taxid != 1  &&
!$found)
                        {
```

```
                    if($tmp->rank ne 'no rank')
                    {
                        $found = 1;
                        $qhash{$tmp->rank}++;
                    }
                    $tmp = $tmp->ancestor;
                }
                    if(!$found)
                    {
                        $qhash{'no rank'}++;
                        $found = 1;
                    }
            }
            else
            {
                $qhash{$taxon->rank}++;
            }
        }
        else
        {
#map taxa that don't match at species level, a level higher in the taxonomy
tree
            my $taxid = $db->get_taxonid($dbdesc);
            my $taxon = $db->get_Taxonomy_Node($taxid);
            $cur = $taxon;
            my $found = 0;
            while(!$found && defined $cur && $cur->ncbi_taxid != 1)
            {
                #only do the ancestor if not found
                if(($names{$read_id}) eq ($cur->scientific_name))
                {
                    if($cur->rank eq 'no rank')
                    {
                    my $tmp = $cur;
                    #print $cur->scientific_name."\n";
                        while(defined($tmp) && $tmp->ncbi_taxid
 != 1  && !$found)
                        {
                            if($tmp->rank ne 'no rank')
                            {
                                $found = 1;
                                $rankhash{$tmp->rank}++;
                            }
                            $tmp = $tmp->ancestor;
                        }
                        if(!$found)
                        {
                            $rankhash{'no rank'}++;
                            $found = 1;
                        }
                    }
                    else
                    {
                        $found = 1;
                        $rankhash{$cur->rank}++;
                    }
                }
                else
                {
                    $cur = $cur->ancestor;
                }
            }
            if(!$found)
            {
                print "$read_id $seq_id ".$dbdesc."\n";
                $rankhash{'no rank'}++;
            }
        }
    }
}
print "Number of reads with exact database matches\t:" . $corr . "\n";
foreach my $key ( keys %qhash )
{
    print "Number of reads correctly assigned at $key level\t:"
      . $qhash{$key} . "\n";
}
```

```
foreach my $rank ( keys %rankhash )
{
    print "Number of reads at $rank level\t:" . $rankhash{$rank} . "\n";
}
sub match_ids
{
      my $inseq = Bio::SeqIO->new(-file => $pppinput, -format =>'fasta');
      while(my $seq = $inseq->next_seq())
      {
            if(exists $names{$seq->display_id()})
            {
                  my $wait;
                  if($seq->desc() =~ /^(\S+)\s/)
                  {
                        $id_list{$seq->display_id} = $1;
                  }
            }
      }
      return %id_list;
}
sub check_inputs
{
      my ( $infile, $pppinput, $indexfile, $usage ) = @_;
      if ( $help || !($infile) || !($pppinput) || !($indexfile) )
      {
            usage();
            exit;
      }
      unless ( -e $infile )
      {
            print "File \"$infile\" doesn\'t seem to exist!!\n";
            exit;
      }
      unless ( -e $pppinput )
      {
            print "File \"$pppinput\" doesn\'t seem to exist!!\n";
            exit;
      }
       unless ( -e $indexfile )
      {
            print "File \"$indexfile\" doesn\'t seem to exist!!\n";
            exit;
      }
}
sub usage
{
    print STDERR <<'USAGE';
       Usage:[options]
      i|infile:     path to file from the ppp output
      p|ppp_input_file: path to the original ppp input reads file
      d|indexfile:      path   to   the   taxonomy   index   file(indexed   by
bp_index.pl)
      h|help: print this help
Example:
perl      ranks_filter.pl      -i      /opt/apps/ppp-backend/scratch/91/Step-0-
refmicrobial.fna_hits.fna  -p  ~/apps/data/simrefmicrobial_microbial_reads.fna
-d /opt/apps/ppp-backend/db/refmicrobial.fna.idx
USAGE
    exit;
}
```

iv) pseudo_db.pl

```
#!/usr/bin/perl

#pseudo_db.pl creates "pseudo database" out of an annotated database in fasta
format,by removing sequences "seen" in the input file (fasta format) and
writes the remaining sequences to another file which can then be treated as a
new database (with formatdb command).

use Bio::SeqIO;
use strict;
use warnings;

my $readsfile = shift;
my $database = shift;
my $dbsequence = shift;

my $in = Bio::SeqIO->new(-file=>$readsfile, -format=>'fasta');
my %accession;

while(my $seq = $in->next_seq())
{
        my $id = $seq->id;
        if($seq->desc =~ /^(\S+)\s/)
        {
              $accession{$1} = $id;
        }
}

my $db = Bio::SeqIO->new(-file=>$database, -format=>'fasta');
my $pseq = Bio::SeqIO->new(-file=>">$dbsequence", -format=>'fasta');
while(my $dbseq = $db->next_seq())
{
        next if(exists $accession{$dbseq->id});
        $pseq->write_seq($dbseq);
}
```

v) get_accesion.pl

```
#!/usr/bin/perl



#get_accession.pl takes an array of readids from the pipeline and searches
#though the corresponding input reads file to obtain the accession numbers,
#then use this to go through the respective database and grab the sequences--
which will be used as input to the readsim

use Bio::SeqIO;
use Getopt::Long;
use strict;
use warnings;
my ( $infile,$readsfile,$database, $sequence, $help );

GetOptions(
    'i|input=s'    => \$infile,
    'r|readsfile=s' => \$readsfile,
    'd|database=s' => \$database,
    's|sequence=s'=>\$sequence,
    'h|help' => \$help
);

check_inputs($infile,$readsfile,$database,$sequence,$help);

my $in = Bio::SeqIO->new(-file=>$readsfile, -format=>'fasta');
open(IN, $infile);
chomp (my @readid=<IN>);
my %accession;
my @found;
while(my $seq = $in->next_seq())
{
      my $id = $seq->id;
      if( $seq->desc =~ /^(\S+)\s/)
      {
            $accession{$id} = $1;
      }
}
foreach my $i(@readid)
{
      push(@found,$accession{$i});
}
my $db = Bio::SeqIO->new(-file=>$database, -format=>'fasta');
my $dbseq = Bio::SeqIO->new(-file=>">$sequence", -format=>'fasta');

while(my $seq_out = $db->next_seq())
{
      foreach my $i(@found)
      {
            if($seq_out->id eq $i)
            {
                  $dbseq->write_seq($seq_out);
            }
      }
}

sub check_inputs
{
    my ( $infile,$readsfile,$database, $sequence, $usage ) = @_;
    if ( $help || !( $infile || $readsfile || $database || $sequence ) )
    {
            usage();
            exit;
    }
    #Does the file exist?
    unless ( -e $infile )
    {
           print "File \"$infile\" doesn\'t seem to exist!!\n";
           exit;
    }
    unless ( -e $readsfile )
    {
      print "File \"$readsfile\" doesn\'t seem to exist!!\n";
      exit;
    }
    unless ( -e $database )
    {
      print "Database \"$database\" doesn\'t seem to exist!!\n";
      exit;
    }
}
sub usage {
    print STDERR<<USAGE;
        Usage:  [options]
            'i|input:path to the input file with the readid's
            'r|readsfile:path to the original reads file
            'd|database:path to the database to comapre against
            's|sequence:file to output sequences for the matching accession
                  numbers
            'h|help' :  display this help
Example:
perl get_accession.pl -i ../data/readid.txt -r /opt/apps/ppp-
backend/data/simrefviral.fna -d /opt/apps/ppp-backend/db/refviral.fna -s
../data/virus_reads.seq

USAGE
exit;
}
```

vi) LCA.pm

```
##LCA.pm shows implementation of the weighted MRCA algorithm alongside the
original MRCA

package LCA;
use strict;
use Bio::DB::Taxonomy;
use Bio::Taxonomy::Taxon;
use Bio::Tree::Tree;
use Bio::TreeIO;

=head1 NAME
LCA.pm: Last common ancestor calculation functions
=head1 DESCRIPTION
This package provides the ability to calculate the last common ancestor of a
collection of reads given taxonomy information
=head1 SUBROUTINES
=cut

=head2 findlcas
      Title   : findlcas
      Function: Finds the LCA for each read in a hash of arrays
      Returns : %lca: A hash of last common ancestors for each read in the
given hash
      Args    : $eqref: a reference to the array of previous equivalent hits
to calculate LCAs for
      $taxondir: The directory containing the names.dmp and nodes.dmp
taxonomy files
      $fxn: The function to call either LCA or MRCA
      Throws  : none
=cut

sub findlcas
{
      my $eqref = shift;
      my $taxondir = shift;
      my $fxn =shift; #which function to call
      my %equiv = %{$eqref};
      my %taxnames;
      my %lca;

      my $db = new Bio::DB::Taxonomy(-source => 'flatfile',
            -directory => $taxondir,
                -nodesfile => "$taxondir/nodes.dmp",
                -namesfile => "$taxondir/names.dmp");

      foreach my $readid(keys(%equiv))
      {
          my @taxon;
          my %descscore;
          foreach my $hit(@{$equiv{$readid}})
          {
               my $desc = $hit->{desc};
               my $bs = $hit->{bitscore};
               my @splitdesc;
               my @taxnames;
               if($desc =~ /\[/) #if we have properly labeled taxon, we'll
use what's in the braces
               {
                    #could have multiple taxons (like in nr) so we'll
have to check them all
                    @splitdesc = split(/\[/,$desc);
                    foreach my $line(@splitdesc)
                    {
                         if($line =~ /^(.*)\]/)
                         {
                              push(@taxnames,$1);
                         }
                    }
               }
               else #if not, we'll just try what they have in the
description line before a comma
               {
                    $desc =~ /^(.+),.*/;
                    $taxnames[0] = $1;
               }
               foreach my $line(@taxnames)
               {
                    my $taxid = $db->get_taxonid($line);
                    if($taxid != 0)
                    {
                         my $taxoncur = $db->get_Taxonomy_Node($taxid);
                         push(@taxon,$taxoncur);
                         $descscore{$taxid} = $bs; #get the bitscores
and the corr. taxid's
                    }
               }
          }
          if(@taxon)
          {
               my $curlca;
               if ($fxn eq "lca")
               {
                    $curlca = calclca(\@taxon); #get the LCA of these
taxons
               }
               elsif($fxn eq "weighted")
```

```
                        {
                                $curlca = calcmrca(\@taxon,\%descscore); #get the
MRCA's of these taxons
                        }
                        else
                        {
                                die "Error, MRCA method undefined";
                        }
                        if($curlca == 1)
                        {
                                $lca{$readid}{name} = 'root';
                                $lca{$readid}{rank} = 'root';
                        }
                        else
                        {
                                my $tmpnode = $db->get_Taxonomy_Node($curlca);
                                $lca{$readid}{name} = $tmpnode->node_name;
                                $lca{$readid}{rank} = $tmpnode->rank;
                        }
                }
        }
        return %lca;
}

=head2 calclca
        Title   : calclca
        Function: Calculates an estimated last common ancestor for an array of
taxonomy nodes
        Returns : $lowpt: The taxonomy id of the calculated last common
ancestor
        Args    : $ref: an array reference of taxonomy nodes
        Throws  : none
=cut

sub calclca
{
        my $ref = shift;
        my @taxons = @{$ref};
        my %anc;
        my $pos = 1;
        my $cur = $taxons[0];
        my $lowpt = $cur->ncbi_taxid;
        #go through the first taxon and map its parents all the way up to the
root
        while(defined $cur && $cur->ncbi_taxid != 1)
        {
                $anc{$cur->ncbi_taxid} = $pos; #pos is the depth into the tree
                $pos++;
                $cur = $cur->ancestor;
```

```
            my $wait = 1;
      }

      $anc{1} = $pos; #map the root node as the highest point
      #for the rest of the taxon nodes
      for(my $i=1;$i<@taxons;$i++)
      {
            my $id;
            $cur = $taxons[$i]; #get the current taxon
            #look at the current node and its parents and find when it
matches with a node in the ancestry of the first
            #will find the lowest common ancestor
            while(defined $cur && !$anc{$cur->ncbi_taxid})
            {
                  $cur = $cur->ancestor;
            }
            if(defined $cur) #if we have a connection, pick it up.  else root
            {
                  $id = $cur->ncbi_taxid;
            }
            else
            {
                  $id = 1;
            }

            #if the newfound ancestor is lower than the current lowest point,
update
            if($anc{$lowpt}<$anc{$id})
            {
                  $lowpt = $id;
            }
      }
      return $lowpt;
}

=head2 calcmrca
      Title   : calcmrca
      Function: Calculates an estimated weighted most recent common ancestor
for an array of taxonomy nodes
      Returns : $wmrca: The taxonomy id of the calculated last common
ancestor
      Args    : $ref: an array reference of taxonomy nodes
              $dref: a reference to the hash of taxonids with their
corresponding descriptions
      Throws  : none
=cut

sub calcmrca
{
        my $ref = shift;
        my @taxons = @{$ref};
        my $dref = shift;
        my %descscore = %{$dref};
        my %bitscore;
        my $tree = undef;
        my $cur;
        my $curbs;
        for (my $i= 0; $i<@taxons; $i++)
        {
                $cur = $taxons[$i];
                $curbs = $descscore{$cur->id};
                if ($cur)
                {
                        if(!defined $tree)
                        {
                                $tree = Bio::Tree::Tree->new(-node=>$cur);
                        }
                        else
                        {
                                $tree->merge_lineage($cur);
                        }
                }
                else
                {
                print  STDERR "Error\n";
                }

        #go through the first taxon and map its parents all the way up to the
root
                while(defined $cur && $cur->ncbi_taxid != 1)
                {
                        $bitscore{$cur->id} += $curbs;#add up the bitscores as you
walk up the tree
                        $cur = $cur->ancestor;
                        my $wait = 1;
                }
        }
        my $root = $tree->get_root_node;
        my @nodeque;
        push (@nodeque,$root);
        my $rootbs = $bitscore{$root->id};
        my $i;
        my $wmrca;
        my $cutoff = sprintf("%.0f",(60/100 * $rootbs));
        #my $cutoff = 2/3 * $rootbs;
        while ($i< @nodeque)
        {
            my @children = $nodeque[$i]->each_Descendent();#get the
descendent of each node
            #loop through the children and push onto the hash if they pass
the %cutoff
            my $count=0;
            foreach my $child (@children)
            {
                  if(($bitscore{$child->id})> $cutoff)
                  {
                        push (@nodeque, $child);
                        $count++;
                  }
            }
            if ($count == 0)#if no child passes the cutoff,return it's parent
            {
                  $wmrca = $nodeque[$i]->id;
                  @nodeque=();
            }
                  $i++;
      }
      my $test;
      return $wmrca;
}

1;
```

The bold text shows the parts in the script that are specific to the weighted MRCA (written during this project).

vii) lcawrap.pl

```
#!/usr/bin/perl
##lcawrap.pl -takes as input the equivalent hits file from the PPP and
returns the different abundances of the read sequences and their taxonomic
ranks(for both mrca and weighted mrca)

use LCA;
use strict;
use warnings;

my $fname = shift; #equiv's file
my $outfile = shift;
my $otherfile = shift;

open(FH,$fname);
open(EXACT, '+>', $outfile); #for the given readid,the descriptions match
open(OTHER, ">>$otherfile"); #file for the  assignments at each level(both
exact/not exact matches)
my %equiv;
my $state = 0;
my $incstate = 0;
my $id;
while(my $line = <FH>)
{
        chomp $line;
        if($state == 0 || $state == 2)
        {
                if($line =~ /^\#(\S+)/)
                {
                        $id = $1;
                        $state = 0;
                        $incstate=1;
                }
        }
        if($line)
        {
                if($state == 1)
                {
                        if($line =~ /^LCA/)
                        {
                                $incstate = 1;
                        }
                        else
                        {
                                $state++;
                        }
                }
                if($state == 2)
                {
                        if($line =~ /^(\S+) bs=\s*(\S*) hsplen=\s*(\S*)
pid=\s*(\S*) expect=\s*(\S*) (.*)$/)
                        {
                                my %tmp;
                                $tmp{acc}=$1;
                                $tmp{bitscore} = $2;
                                $tmp{length} = $3;

                                $tmp{pid} = $4;
                                $tmp{expect} = $5;
                                $tmp{desc} = $6;
                                push(@{$equiv{$id}},\%tmp);
                        }
                }
        }
        if($incstate==1)
        {
                $state++;
                $state = $state % 3;
                $incstate = 0;
        }
}
my %lca = LCA::findlcas(\%equiv,"/opt/apps/ppp-backend/taxon/","lca");
my %mrca = LCA::findlcas(\%equiv, "/opt/apps/ppp-backend/taxon/","weighted");

#compare the two hashes, get the differences in the assignments by the two
algorithms
my $taxondir = "/opt/apps/ppp-backend/taxon";
my $db = new Bio::DB::Taxonomy(
                -source =>'flatfile',
                -directory => $taxondir,
                -nodesfile => "$taxondir/nodes.dmp",
                -namesfile => "$taxondir/names.dmp"
        );
my @lcanames;
my @mrcanames;
foreach my $readid (keys (%lca))
{
        if(($lca{$readid}{name} eq $mrca{$readid}{name}) &&
($lca{$readid}{rank} eq $mrca{$readid}{rank}))
        {
                print EXACT $readid."\t".$mrca{$readid}{name}."\t".
$mrca{$readid}{rank}."\n";
        }
        if(($lca{$readid}{name} ne $mrca{$readid}{name}) ||
($lca{$readid}{rank} ne $mrca{$readid}{rank}))
        {
                        #get the different taxonomic ranks
                push(@lcanames, $lca{$readid}{name});
                push (@mrcanames, $mrca{$readid}{name});
        }
}
my %lcadf;
my %mrcadf;
my %uptree;
for(my $i=0;$i<@mrcanames;$i++)
{
        my $mrca = $mrcanames[$i];
        my $lca = $lcanames[$i];
        my $mrcataxid = $db->get_taxonid($mrca);
        my $lcataxid = $db->get_taxonid($lca);
        my $ltaxon = $db->get_Taxonomy_Node($lcataxid);
        my $taxon =$db->get_Taxonomy_Node($mrcataxid);
        if(defined $ltaxon && defined $taxon)
        {
                my $lcanode = $ltaxon->node_name;
                my $mrcanode = $taxon->node_name;
                if($ltaxon->rank)
                {
                        $lcadf{$ltaxon->rank}++;
                }
                if($taxon->rank)
                {
                        $mrcadf{$taxon->rank}++;
                }
                while(($taxon->rank eq 'no rank') && ($taxon->id != $ltaxon->id))
                {
                        $taxon = $taxon->ancestor;
                        #move up the tree till we hit the same assignment as for
lca
                        $uptree{$taxon->rank}++;
                }
        }
        else
        {
                print STDERR "Warning: Entry is not defined\n";
                print STDERR $mrca."\t$lca\n";
        }
}
#cant close the file, have to read from it later
my %ranks;
seek(EXACT,0,0); #seek to the beginning of the file
while(<EXACT>)
{
        chomp;
        my @lines = split(/\t/,$_);
        my $name = $lines[1];
        my $rank = $lines[2];
        if($rank)
        {
                $ranks{$rank}++;
        }
}

foreach my $key (keys(%ranks))
{
        print OTHER "Reads with exact matches assigned at $key
level\t:".$ranks{$key} ."\n";
}

print OTHER "\n";
foreach my $key (keys(%lcadf))
{
        print OTHER "Reads assigned by old lca at $key
level\t:".$lcadf{$key}."\n";
}

print OTHER "\n";
```

```
foreach my $key (keys(%mrcadf))
{
      print OTHER "Reads assigned by mrca at $key
level\t:".$mrcadf{$key}."\n";
}
foreach my $rank(keys(%uptree))
{
      print OTHER "  "."Reads reassigned at $rank
level\t:".$uptree{$rank}."\n";
}
```