ABSTRACT

Title of Dissertation: COMPARATIVE AND COMPUTATIONAL METHODS FOR MICROBIAL GENOMICS

Derrick Edward Wood, Doctor of Philosophy, 2014

Directed by: Professor Steven L. Salzberg
Department of Computer Science

Through the study of genomic sequences, researchers are able to learn much about the workings of life. As sequencing technology has improved over the past decade, the number of genomes that have been assembled has grown exponentially, and the amount of sequence generated by sequencing machines can easily number in the billions or even trillions of nucleotides for a single project. This rise in the amount of information present requires informatics approaches to correctly and efficiently analyze the data. One common approach has been to use comparative methods, which use sequence similarity to infer functional or evolutionary relationships between sequences. This dissertation uses comparative methods to improve existing records of genomic data, and introduces a novel computational approach to the problem of taxonomic sequence classification.

The first part of this dissertation uses two approaches involving pairwise and multiple sequence alignments to find and correct errors in the public records of microbial genomes. Through alignment to sets of genes with known function, we

show that thousands of genes have been mistakenly omitted from our public records. Our analysis of these genes shows a tendency for short genes to be omitted, and reveals that genes are more frequently omitted by organizations with less experience in annotating genomes. We also use multiple alignments of protein sequences to improve the annotation of start positions of genes, in some cases restoring hundreds of nucleotides to the genes' records. Through analysis of our results, we also found a link between a high use of rare start codons and a high rate of erroneously annotated start sites.

The final part of this dissertation presents a method involving exact alignment of short sequences to perform rapid taxonomic sequence classification. By using the existing concept of minimizers to increase CPU cache utilization, we have created a tool capable of performing taxonomic classification with a sensitivity that is comparable to existing methods, a precision that surpasses all existing methods, and a speed that is over 900 times faster than the fastest existing classification approach.

COMPARATIVE AND COMPUTATIONAL METHODS FOR MICROBIAL GENOMICS

by

Derrick Edward Wood

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2014

Advisory Committee:

Professor Steven Salzberg, Chair
Professor Mihai Pop, Co-chair
Professor Alan Sussman
Professor Héctor Corrada Bravo
Professor Najib El-Sayed, Dean's Representative

Acknowledgements

The support of my advisor, Steven Salzberg, has been invaluable to me over the past four years. Without his guidance, patience, and experience, I would not have been able to successfully navigate the challenges presented by graduate school and to become a successful scientist.

I met both Mihai Pop and Alan Sussman prior to my first day of graduate classes, and throughout my time here their advice has consistently improved my work. In particular, working with Alan over my first years here was a great experience, and I am grateful for the opportunity he gave me to have such an impact on future classes of computer science students here.

Several of my fellow students in the Center for Bioinformatics and Computational Biology have helped me over the past four years. In particular, Daehwan Kim, Ben Langmead, Dave Kelley, Chris Hill, Ted Gibbons, Joe Paulson, Joyce Hsiao, and Kwame Okrah have all helped me to improve my research; to better understand biology, computer science, and statistics; and to deal with the day-to-day problems of graduate school.

I would also like to thank three CBCB alumni – Adam Phillippy, Todd Treangen, and Sergey Koren – who are now currently at the National Biodefense Analysis and Countermeasures Center. They have all helped me in the development of my software, and given it critical endorsements that have made a great difference in my career.

Outside of CBCB, my friends in the Computer Science department – especially Brandon Wilson, Patrick Roos, João Soares, and Leo Claudino – have made my time since returning to Maryland much more fruitful and enjoyable than it otherwise would have been.

While I was a student at the University of Texas at San Antonio, Kay Robbins, Hugh Maynard, and Cathy Key all helped me in obtaining opportunities to begin research, and to begin teaching students. The experience in teaching and tutoring I gained there provided me with much of the practice I needed to be a successful speaker in my graduate career. I would also especially like to thank Hugh for being the first to suggest – and for doing so repeatedly – that I pursue a doctorate degree.

And finally, I need to thank my parents for encouraging me through all the times in my youth when I expressed a desire to learn more about nearly anything. I consider myself very fortunate to have had their love and support throughout my life.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Background

1.1.1 DNA and sequencing technology

Within the cells of all living organisms are one or more molecules of deoxyribonucleic acid (DNA), and each molecule of DNA consists of two strands of nucleotides. The 4 nucleotides of DNA are adenine (A), cytosine (C), guanine (G), and thymine (T). The two strands of nucleotides are joined together by hydrogen bonds between the nucleotides, and a bonded pair of nucleotides is often referred to as a base pair (bp). In normal conditions, adenine will only bond with thymine, and cytosine will only bond with guanine; this means that the sequence of nucleotides on one strand of DNA can be determined by the sequence on the other strand. The sequence of an organism's DNA is the organism's genome, and that sequence provides genetic information that guides the organism's development.

Analysis of an organism's genome often involves whole genome shotgun sequencing, where DNA is broken into many fragments. Each fragment is then examined by a sequencing machine, which reports the sequence of the fragment with an error rate that varies between machines [1]; these sequences are called reads. The read length that can be obtained varies depending on the type of sequencing machine, but is measured in the hundreds (or in some cases, thousands) of base pairs. In contrast, most known bacterial genomes are 2-6 million base pairs long, and the human genome contains 3.3 billion base pairs.

This means that each read represents only a small fraction of the genome that the read came from.

1.1.2 Protein-coding genes and open reading frames

Within a bacterial cell, molecules of ribonucleic acid (RNA) are created from various parts of the DNA. These RNA molecules, known as transcripts, are nearly identical in sequence to the DNA they are created from, with the nucleotide uracil used in place of thymine. Some of these transcripts are messenger RNA (mRNA) molecules, which are then processed by a ribosome to produce a protein through a process known as translation. The regions of the DNA that are eventually translated are known as protein-coding genes. There are other types of genes, including ribosomal RNA (rRNA) and transfer RNA (tRNA), but when used without qualification the term "gene" most often refers to protein-coding genes.

During translation, the ribosome scans the mRNA three nucleotides at a time, and produces a chain of amino acids, with one amino acid being added to the chain for each nucleotide triplet encountered. These triplets, also known as codons, effectively provide instructions to the ribosome on how to proceed with the translation process. 61 of the possible 64 codons will cause the recruitment of an additional amino acid to the growing chain, while 3 stop codons (UGA, UAA, UAG) cause the translation to terminate when they are scanned by the ribosome. Translation typically only begins at AUG, GUG, or UUG codons, with AUG codons representing the large majority of known start codons.

Because the mRNA is read in triplets, any given strand of RNA has three possible reading frames (DNA has six, as the reverse strand must also be considered). For example, the sequence AGGCC can be read starting at any of the first three characters, resulting in the first codon encountered being AGG, GGC, or GCC. Depending on which reading frame is used, the translation of a given RNA or DNA sequence will be very different. An open reading frame (ORF) is a region starting immediately following a stop codon and ending with the first downstream stop codon in the same reading frame. With the exception of a very small number of genes that span multiple reading frames due to ribosomal slippage ("frameshifted" genes), all protein-coding genes in bacterial genomes are found within open reading frames. In addition, all protein-coding genes end at the downstream end of an open reading frame; the start of these genes is not as easy to determine, as the possible start codons can serve as internal codons as well.

1.1.3 Gene finding and annotation

For those genomes that have been fully sequenced and assembled, a common research goal is to analyze the genes present in the genome. Several programs have been developed to find genes in a genome, often building a compositional model of protein-coding DNA in the process. For example, Glimmer [2–4] uses Markov models to represent the sequences often found in a genome's coding DNA, and then uses this model to score all open reading frames in the genome. Other programs, such as Prodigal [5], use hexamer (6 bp substrings) frequencies to create a coding DNA model.

To create a model of coding DNA, a program must have a large amount of DNA sequences that are very likely to be coding DNA. This can be done by having a user supply a set of known genes, or by examining the open reading frames in the genome and selecting those that are so long that it is unlikely that the reading frame does not contain a gene. To increase the number of selected long ORFs that contain a gene, other statistics such as codon frequencies can be used to inform the selection process; Glimmer does this with its long-orfs program, selecting ORFs with a codon distribution that approximates the "universal" bacterial codon distribution [4].

Once a set of genes is obtained via a gene prediction pipeline, the genome can be annotated to note the locations of these genes on the genome. Further analysis can involve attempting to determine the function of these genes; the locations and functions of the genome's protein-coding genes make up the large majority of a microbial genome's annotation. Although laboratory experiments can help determine a gene's function, the large number of genomes that are sequenced makes this approach unrealistic for most genes. Instead, a gene *G* is often compared to a large database of genes, and if *G*'s sequence is very similar to a gene with known function, that function is considered to be *G*'s function as well.

1.1.4 Sequence alignment

To explore the relationships between two or more genomic sequences, it is often necessary to gain a measure of the similarity between them by aligning the sequences. Using a dynamic programming approach, an optimal alignment between two sequences can be found with respect to a given scoring scheme.

Such pairwise sequence alignments can be global in nature, attempting to align each sequence end-to-end [6], or can be local, allowing the alignment to begin and end in the middle of the two sequences [7]. These dynamic programming algorithms have a computational complexity of $O(mn)$, where m and n are the lengths of the two sequences being aligned. In cases where large sequences must be aligned, the memory and runtime requirements of these algorithms exceed what is reasonable, and other methods exist that reduce these requirements at the expense of losing the optimality guarantee of the dynamic programming algorithms.

A commonly used and widespread program to align a query sequence against a large database of sequences is BLAST [8, 9]. The BLAST algorithm for nucleotide alignment divides the query sequence into overlapping strings of length k (or k-mers), and finds all known locations of these k-mers in the database using exact matching. Each occurrence of a query sequence k-mer in the database (an exact alignment) serves as a seed for the next step of the algorithm, where the seed alignment is extended in both directions until extension would cause a decrease in alignment score. Alignments that exceed a specified E-value threshold are then reported. An alignment's E-value is the number of alignments that would be expected to score as well or better than the given alignment, if a random query sequence and database of the same lengths were used to perform the search. A similar approach exists for protein sequence alignments, but certain differences from the query sequence k-mers are allowed to be present in the seeds. In addition to BLAST, a similar program BLAT [10] performs alignments even faster,

through the use of an index of non-overlapping k-mers stored in physical memory, but it is less sensitive than BLAST.

Alignment of reads to a known genome is also a common operation, and the gigabases of data that are generated by current sequencing technology cannot be efficiently by either BLAT or BLAST [11]. To address this, tools such as MAQ [12], SOAP [11], Bowtie [13, 14], and BWA [15] were created, utilizing different indexing strategies to make read alignment faster. The most successful of these has been the use of the Burrows-Wheeler transform [16] and the FM index [17], used in both Bowtie and BWA.

In addition to pairwise sequence alignment, multiple sequence alignments are possible, allowing researchers to examine the relationships between several homologous regions at the same time. Programs such as Clustal W [18] and MUSCLE [19] can create these multiple sequence alignments for both nucleotide and protein sequences.

1.1.5 Comparative genomics

Common regions between two genomes can indicate information about the relationship of the regions to those genomes, and information about the relationship between the two organisms. If a region in one genome is similar to a region in another genome, but the two genomes are largely different, then the regions are likely to have maintained their similarity under evolutionary pressure. The evidence of this pressure means that the regions likely serve a necessary and similar function for their respective organisms [20, 21].

In closely related genomes, we expect high similarity to exist in many regions of the genomes, due to the lack of evolutionary time for mutations to occur and cause large differences between the genomes. This property means that, without knowledge of the origins of the two sequences, high similarity does not imply functionality (or even conservation). However, high similarity can still imply a close evolutionary relationship between the two sequences. This relationship allows us to use BLAST searches to accurately assign taxonomic labels to sequences of unknown origin by using the highest-scoring match as the label [22, 23].

1.1.6 Metagenomics

While most genomic studies have involved the use of DNA that is known to be from a single species (and often from the same strain), the past decade has seen a rise in metagenomic studies, where all the DNA in an environmental sample is sequenced, without an attempt to culture specific members of the sample. These studies have allowed researchers to obtain data from the genomes of organisms that cannot be cultured in a laboratory setting. Metagenomic studies also allow researchers to learn about the genes and taxonomic distribution of the populations in various environments such as the guts of obese and lean humans [24] and acid mine drainage [25].

1.2 Motivation

In 1995, the genome of *Haemophilus influenzae* was published [26], marking the first time a bacterial genome had been sequenced. Less than 15 years later, in late 2009, the 1,000th bacterial genome was completed [27]. Not even five years after

that event, the number of completed genomes has nearly tripled: in early 2014, over 2,700 completed bacterial genomes exist with nearly 10,000 more genomes available as draft assemblies. This sharp rise in the number of completed genomes is in large part due to the increasing throughput of sequencing technology, and the decreasing cost of sequencing a genome with the high coverage needed to perform an assembly. Although these increases in completed genomes and throughput provide much more data than was previously available, and so can be of great help to researchers, there are issues that accompany these increases. Two of these issues affect comparative methods' accuracy and efficiency, and this dissertation addresses each of these problems through the use of comparative methods and novel algorithms.

1.2.1 Errors in genome annotation

The large number of completed genomes makes comparative genomics approaches more powerful, because these approaches will have a much richer set of references to compare against. As the growth of the public genome databases continues, researchers will be able to gain further insight into the mechanisms through which life works.

An implicit assumption in comparative genomics, and in any work performed that utilizes a reference database of genomic information, is that the reference is correct. The NCBI Reference Sequence (RefSeq) database [28, 29] was created with the goal of providing a set of genomes where the sequences and corresponding annotations had been manually curated and validated. This curation would provide a database that could be used by a researcher without requiring

further validation of the reference data before beginning the analysis that is of interest to the researcher. As the number of completed genomes has increased, many of these new genomes are entered into the RefSeq database without undergoing manual curation. Given the time, labor, and material costs in performing such curation, it is unlikely that many new genomes will be individually validated. Without this curation, the sequences in RefSeq now contain annotations that are largely unmodified from those supplied by the researchers who deposited the sequence into the GenBank database (from which many of RefSeq's records originate). This effectively means that the annotations are not at a uniform standard of quality and cannot be relied upon without further validation, which in large part defeats one of the main purposes of the RefSeq database.

Computational methods to evaluate and improve the accuracy of existing annotations are necessary because the growth of sequenced genomes has outpaced the ability of researchers to perform the laboratory work needed to provide experimental evidence to support the annotations. The vast majority of annotations are performed solely via computational and comparative methods, using gene finding programs and sequence similarity searches to locate the genes in a genome and discover the functional properties of these genes. Existing gene finding programs are quite sensitive: for example, Glimmer is able to correctly predict the existence of 99% of genes in microbial genomes [4]. However, gene finding programs often have tunable parameters, and while these can be used to improve a program's performance, they can also be misused to produce gene

predictions that are far from accurate. When inaccurate annotations are produced by researchers and deposited into public databases such as GenBank and RefSeq, these annotations can confound future research, and the errors in these annotations can propagate into future annotations. Therefore, it is imperative that incorrect information in these databases be found as quickly as possible and corrected [30]. In an effort toward finding and correcting erroneous annotations, this dissertation addresses this problem by contributing two pipelines for genome annotation analysis, both of which use a strategy of alignment to other sequences to support their results.

Chapter 2 presents a pipeline that is used to determine if an existing annotation is missing genes that are truly present in the genome. In reviewing the pipeline's results, over 10,000 genes are wrongly omitted from the genomes deposited in GenBank, including many that may be of interest to pharmaceutical researchers. Using commonly available tools such as Glimmer [3] and BLAST [9], this pipeline can be implemented by any laboratory performing genome annotation, and should enable such laboratories to avoid missing genes for which ample evidence supports their annotation.

Related to the problem of gene finding is the problem of start site annotation. Whereas the three stop codons always terminate translation and so can serve as a clear end site for a gene, the possible start codons can also occur in the interior of a gene. Gene finding programs typically have to incorporate upstream signals, such as ribosome binding sites, in order to achieve a start site prediction accuracy of over 90% [31]. Even with this extra information, however, start site prediction

accuracy lags behind overall gene prediction accuracy. Correctly identifying the start site is important because an incorrect start site annotation can give the impression that large portions of a protein's sequence are not actually part of a gene, when these wrongly omitted portions can provide necessary functionality to the organism. Likewise, wrongly included protein sequence can mislead researchers into believing a protein has a function that it does not truly have.

In Chapter 3, I present Phantim, a pipeline using multiple alignments of proteins to address the problem of start site annotation for protein-coding genes. Phantim uses these multiple alignments to search for conservation-based evidence for the selection of a given start site. By requiring strong conservation to make any prediction, Phantim reports a small set of genes for which we estimate over 99% are correct. After examining the differences between existing annotations and Phantim's predicted annotations, I developed an easily-calculated test to determine if an existing annotation was incorrectly using rare start codons in a way that was likely to lead to widespread incorrect start site annotation.

1.2.2 Computational burden of increased sequencing throughput

The ability of today's sequencing machines to generate millions of sequences per sequencing run has enabled many new studies. However, analysis of these millions of sequences can require enormous amounts of computational resources. Requiring researchers to wait several days, if not weeks or months to analyze their data can cause those researchers to not even consider various avenues of research, as those avenues would be considered unreasonably expensive to pursue. It was this problem that motivated the development of tools like Bowtie [13] and BWA

[15] to rapidly align reads to a reference genome, and these tools have proven themselves extremely useful in the years since their introduction: the 2009 Bowtie paper has seen over 3,000 citations in the 5 years since its publication.

In metagenomics, taxonomic sequence classification is the problem of assigning a taxonomic label (e.g., a genus or species name) to a DNA sequence. This is related to the problem of aligning reads to a reference genome; in fact, one method for doing taxonomic classification involves aligning a read against a large set of reference genomes using BLAST. However, this has proven to be computationally intensive and time consuming, with even the fastest BLAST program, Megablast, only able to process less than 10,000 reads per minute. Using such an approach renders downstream research much more difficult, as the classification step would take days for a common sample size of 30 million sequences. Larger samples, such as the over 20 trillion base pairs of sequence generated by the Human Microbiome Project (HMP) [32], would require enormous amounts of computational processing to obtain classification results in a reasonable amount of time. To process this large amount of sequence, the HMP turned to the abundance estimation program MetaPhlAn [33].

Abundance estimation programs such as MetaPhlAn or MetaPhyler [34] work by aligning reads to a small database containing only clade-specific or "marker" genes. Alignment against these smaller databases allows the programs to run much faster than alignment against a set of reference genomes: MetaPhlAn can process over 400,000 reads per minute. But this speed comes at a cost: only a small subset of reads (those that are from the regions covered by the marker

genes) can be classified. Thus, abundance estimation programs use this subset to report population abundance estimates rather than attempt to classify all reads in a sample.

Ideally, it would be possible to classify the large majority of reads in a metagenomic sample, and do so quickly. In Chapter 4, I introduce Kraken, a program that accomplishes both of these goals through the use of k-mer matching and a pre-computed database. Kraken's accuracy is comparable to the fastest BLAST program, while its speed exceeds other classifiers by multiple orders of magnitude. The k-mer matching of Kraken is the element that gives Kraken a majority of its speed, as the database allows a k-mer to be compared against a set of n genomes in an $O(\lg n)$ time operation. In addition, Kraken is made even faster through the application of the existing minimizer concept [35] in a novel manner that allows increased locality and CPU cache utilization. Kraken was released to the public in 2013, and the paper describing its methods was published in March 2014.

2 Discovery and analysis of unannotated genes in microbial genomes

This chapter describes work performed with Henry Lin, Ami Levy-Moonshine, Rajiswari Swaminathan, Yi-Chien Chang, Brian P. Anton, Lais Osmani, Martin Steffen, Simon Kasif, and Steven L. Salzberg. In this work, we developed a pipeline for discovering genes that had mistakenly been omitted from existing genome annotations. Using a database of genomic data that includes experimentally verified genes and many other sources of information, we provided evidence that a large majority of the missed genes we found were indeed true genes. We also investigated the causes for the missed genes' omission, focusing on issues such as gene length and the experience of the centers performing the genome annotations. This work was published in *Biology Direct* in October 2012 [36].

My contribution to this work included development and implementation of the missed gene discovery pipeline as well as analysis of the causes of the missed genes. Henry Lin helped in developing the pipeline and performing the missed genes cause analysis. Ami Levy-Moonshine, along with Rajiswari Swaminathan and Yi-Chien Chang, contributed the analysis of the missed genes with the COMBREX database. All of the authors helped in the design of this study, and in the preparation of the text.

2.1 Background

Bacterial gene identification has improved considerably since the first bacterial genome, *Haemophilus influenzae* Rd KW20, was sequenced in 1995 [26].

Bacterial genome annotation is usually done through an automated process that identifies most genes very accurately, with sensitivity sometimes exceeding 99% [4]. As a result of the declining cost and increasing ease of genome sequencing in recent years, 1,699 prokaryotic genomes (113 archaea and 1,586 bacteria) have been completely sequenced, and nearly 5,000 more draft genomes have been deposited in public archives. The genomes of most of these species have been sequenced and annotated by relatively large sequencing centers, but many smaller centers and even individual laboratories have contributed some genomes as well. The continuing reduction in the cost of sequencing suggests that the trend towards sequencing by small laboratories will increase substantially in the future. Many of these smaller laboratories do not have substantial in-house bioinformatics expertise. The annotation process can vary greatly from one center to the next, and even within a center it varies from year to year, with different programs used for gene finding, alignment, and assigning gene names.

Several different gene finding programs have been used over the years, including Glimmer [2–4], GeneMark [37, 38], and others, and each of these programs has itself gone through different versions that produced changes (mostly improvements) in their performance. The one program that has been consistently used over the years and across all species is BLAST [8], which is most commonly used to search predicted protein-coding genes against an ever-growing database of previously reported proteins. Significant sequence similarity with a protein in another species is strong evidence that the predicted protein is genuine, especially

if the target species is evolutionarily distant. BLAST searches for homologous sequences have long been the gold standard of evidence for gene prediction.

As a result of the various methodologies, annotation is not very consistent between prokaryotic genomes, even for different strains of the same species, unless the annotation was all performed at the same sequencing center and within a relatively short time period. Genes can be missing from the annotation of some strains and present in others; their start codon positions can vary widely, yielding genes with apparently different lengths; genes can be labeled as pseudogenes or not, depending on the conventions used by the original annotators; and genes can be annotated on the wrong strand or the wrong reading frame. Other differences in annotation programs or in the parameters used to run them can also influence the set of genes found, for example by omitting genes below an arbitrary length threshold. Addressing these inconsistencies should improve current methodologies for prokaryotic annotation.

Obtaining perfect annotation for a bacterial genome is still beyond our reach, even though methods continue to improve. Thus it is reasonable to assume that there are genes missing in current annotations. Finding even some of those genes that have been omitted and correcting other flaws will have direct effects on our biological understanding of the species in question. Many of the missed genes can be associated with specific biochemical functions, thus contributing to our knowledge of the species' molecular machinery. In some cases, for example when a missed gene in a potentially pathogenic organism is associated with antibiotic

resistance, identifying it can help us better understand the causes and treatments of infections.

For this reason, we took a very broad look at all the completely sequenced prokaryotic genomes to determine how many likely genes are simply missing from the annotation and are easily found with our proposed pipeline. We focused on this question because protein sequence homology, as measured by BLAST, provides a highly reliable, consistent tool for identifying missing genes. Although gene identification is generally much better for prokaryotes than for eukaryotes (whose gene structure is much more complex), many genes are nonetheless missing entirely from finished, published genomes.

In this work, we do not intend to find the entire set of missing genes but instead to demonstrate a relatively simple way to find a large set of likely missed genes. In addition to identifying thousands of missing genes, we also provide some possible explanations for their omission, and provide analysis and information about each missed gene in a publicly available database (see details below). It is important to note that there are additional issues with gene annotation that are beyond the scope of our work, but should be addressed by the annotation community. In addition to missing genes, we found many other inconsistencies, including genes of varying lengths and with clearly incompatible names, but resolving those inconsistencies is much more difficult, often involving manual curation.

Another problem that may arise in genome annotation is the problem of overannotation, the incorrect annotation of ORFs in genomes which are not true

genes. Unfortunately, there are no reliable methods for definitively determining that a particular ORF is not a true gene. Experiments can show that an ORF is not expressed under certain conditions, but showing that an ORF is not expressed under all conditions is not feasible. There have been attempts to quantify overannotation by examining the length distribution of annotated genes, and showing that the length distribution of all annotated genes does not match the length distribution of only annotated genes with supporting homology to known genes [39]. Since it is impossible to precisely determine the extent of overannotation in a genome, we focused on finding unannotated genes that had considerable computational support for their protein-coding nature and were able to be found with freely available and commonly used software.

Our approach to finding missed genes involves using a combination of Glimmer to find ORFs that were likely to be protein-coding genes, which were not present in the existing annotations from GenBank [40], and then using BLAST to find genes that had significant sequence similarity to previously annotated genes. After ruling out potential pseudogenes, we were left with 52,605 ORFs, which we called candidate missed genes, from 1,474 completely sequenced prokaryotic genome annotations. Although many of the candidate missed genes had sequence similarity to proteins with functional annotations, providing strong evidence that they were true missed genes, there were also many candidate missed genes with sequence similarity to only hypothetical proteins, which are putative proteins with no known function. These hypothetical proteins may have only been predicted by gene prediction programs without additional evidence to support a claim that they

are true genes. For those candidate missed genes with homology only to hypothetical proteins, we needed additional information to determine if they were indeed genes.

To this end, we make use of the newly available resource COMBREX [41], which is an online database [42] containing functional predictions and phenotype information for more than 3 million microbial genes (see Methods for more details). Using this knowledgebase, we were able to assign each gene to a COMBREX support level, which helps estimate how likely each potential missed gene found is to be a true protein-coding gene. For instance, if a candidate missed gene has a homolog in COMBREX that has been experimentally cloned and tested for function, we assign the gene to the strong COMBREX support level, and our confidence that this candidate missed gene is a fully functional, protein-coding gene increases. In our analysis, most of the candidate missed genes that share sequence similarity with non-hypothetical proteins were found to have strong COMBREX support. For the candidate missed genes that only have sequence similarity to hypothetical proteins, a significant number were also found to have evidence from COMBREX showing that they are likely protein-coding genes. Additionally, we were able to use COMBREX to assign functional and phenotype information to many of the missed genes we identified.

Previously, Warren, et al. [43] performed a similar study to find missing genes in 1,297 annotated prokaryotic chromosomes and plasmids in RefSeq. They reported over 38,000 of what they termed “absent annotations”, or “putative genes by similarity to currently annotated genes” [43]. Our criteria for determining

candidate missed genes roughly match their criteria for determining "absent annotations," although we find more candidate missed genes, as we analyze more genomes. Warren, et al. also found 1,000 additional "missed genes," which are unannotated ORFs with sequence similarity to other ORFs in other distant species.

Our analysis differs from Warren, et al. by going further to determine the subset of candidate missed genes that have strong support to be actual protein-coding genes, and analyzing them with COMBREX. We use COMBREX both to provide further evidence of the gene's protein-coding nature, and to draw attention to candidate missed genes with important phenotypes that might be of special interest. We also suggest possible reasons for the omission of missed genes, which can be put into practice to improve future annotation efforts.

Although the analysis of Warren, et al. may find more missed genes, as they analyze every intergenic ORF, our focus on predicted yet unannotated genes is able to find a comparable number of missed genes while expending less computational effort in the task of searching for homologous genes. We found that Glimmer predicted over 97% of genes in RefSeq bacterial genome annotations, suggesting that our approach will find a substantial number of missed genes without needing to search all intergenic ORFs. Even though a small number of missed genes may be overlooked by our approach, our goal with this study was not to find all missed genes, but rather a large subset thereof that could be easily found through existing tools. This large subset of missed genes, while not complete, should nonetheless be useful for the research community.

Currently, the entire set of missed genes is accessible online [44] in the form of downloadable lists of genes and sequences, divided by some basic criteria. Eventually, these genes will be fully integrated into COMBREX, which will allow one to search for a particular gene based on specific attributes, view information associated with the gene, and utilize other functionality available from COMBREX.

2.2 Methods

2.2.1 Overview of Analysis

The process used to identify missed genes is summarized in Figure 1. We began by looking at a set of 1,574 prokaryotic chromosomes with GenBank annotations from 1,474 completely sequenced genomes. We used Glimmer3 [4] and a few consecutive filtering steps to identify a set of candidate missed genes. These genes were further separated into two distinct subsets based on the nature of the homologs of each individual candidate missed gene. A candidate missed gene that shared significant sequence similarity (based on BLAST similarity scores) with a known gene with non-hypothetical annotation was termed a named missed gene. Named missed genes are very likely to be missed genes, given their homology to a known protein with functional annotation. The remaining candidate missed genes were termed hypothetical missed genes. These two phases of the pipeline are described in more detail in the first part of this Methods section.

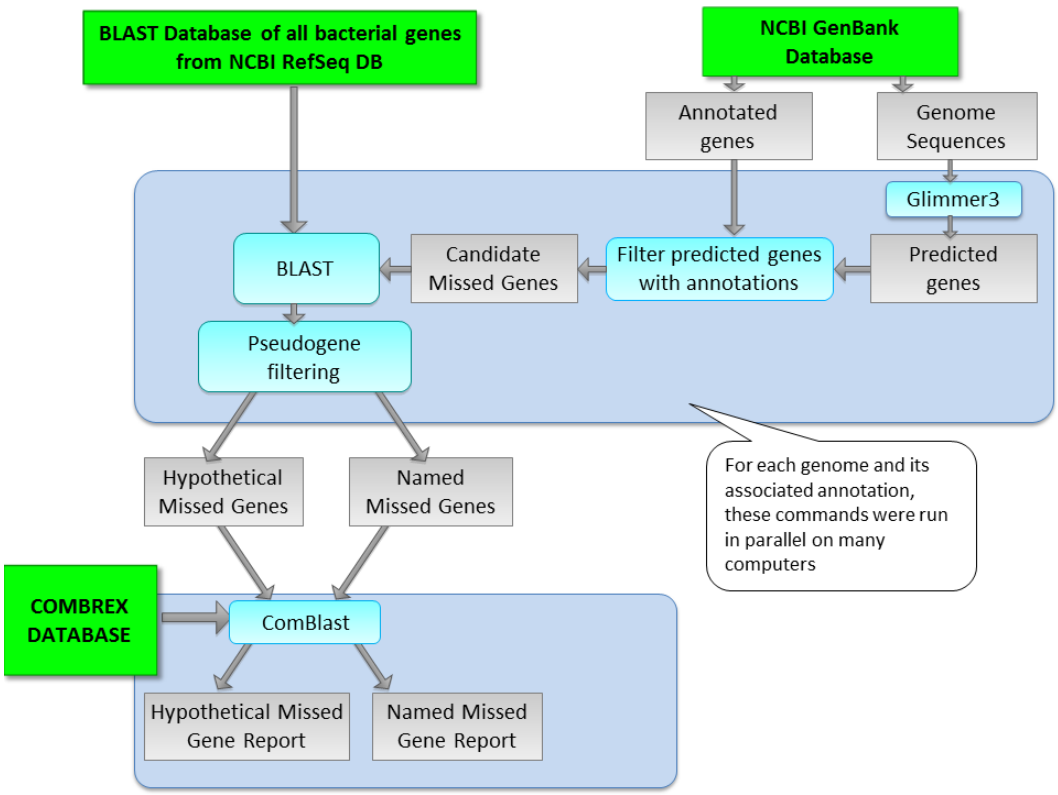

Figure 1. Data flow through our analysis pipeline. Annotations and sequences were obtained from GenBank, and all sequences were processed with the Glimmer 3 gene finder to obtain gene predictions. Sets of predicted genes were filtered to exclude annotated genes and pseudogenes to obtain a set of candidate missed genes. These predicted genes were input as queries to BLAST against a database of all bacterial genes in RefSeq. Predicted genes were then designated as named missed genes or hypothetical missed genes, based on if they had a significant alignment to a non-hypothetical protein, or only aligned to hypothetical proteins, respectively. Each of these two sets were further analyzed by ComBlast, which uses BLAST and the COMBREX database to associate genes with additional attributes, such as experimentally determined function, 3D structure, conservation and phenotype information and assign a COMBREX support level to each potential missed gene.

In the next step, we mined COMBREX for functional information about the candidate missed genes. We were able to assign functional and phenotype information to many of the missed genes using ComBlast, a tool that associates the query missed genes with existing data in COMBREX through sequence similarity methods. Based on the data stored in COMBREX, we were also able to assign COMBREX support levels to each missed gene, indicating whether or not

it was a true protein-coding gene. Both COMBREX and its use in our analysis are described later in this section.

2.2.2 Preprocessing and Filtering

As shown in Figure 1, we first downloaded all GenBank files (files with a .gbk extension) corresponding to prokaryotic genomes from NCBI [40] on May 16, 2011. We then removed any GenBank files that represented plasmids so that we could focus on the main chromosomes. We decided to exclude plasmids from our analysis because gene prediction methods do not work well on very small sequences, and plasmids typically represent only a small percentage of a genome. We also removed 39 GenBank files that did not contain any annotations at all, presumably because the annotations have not yet been completed. This was an important filtering step as it eliminated around 100,000 of the missing genes we originally thought we found.

Similarly, we took care to remove any draft genomes, as Glimmer may make gene predictions which extend beyond the end of a contig or extend into a region containing many ambiguous nucleotides in the sequence. Since we were unsure if the new genes we found for these genomes were true missed genes, we decided instead to exclude draft and incomplete sequences that we had initially found in the list of "complete" genomes at NCBI. We excluded 36 genomes containing the phrases "draft" or "nearly complete" in the header line, or whose sequences contain more than 10 distinct locations with at least 5 consecutive ambiguous nucleotides. After performing these initial filtering steps, we were left with 1,574

GenBank files, for which we used Glimmer3 and BLAST to find our final set of candidate missed genes.

2.2.3 Gene Prediction and Finding Missed Genes

For each of the 1,574 chromosomes, we generated a set of *ab initio* gene predictions by running Glimmer3 with the g3-iterated.csh script, yielding roughly 4.94 million predicted genes. The only modifications we made to the standard options in the g3-iterated.csh script were for 35 chromosomes, mostly from the genus *Mycoplasma*, that use a non-standard genetic code, where we ran Glimmer3 with the option "-z 4" to set the stop codons used by Glimmer3 to be only TAG and TAA.

We then compared the predicted genes with the annotations in the original GenBank file to find roughly 350,000 predicted genes missed in the GenBank annotation. A predicted gene was considered present in the original annotation if its 3' end was shared with any annotated CDS in the GenBank file. Predicted genes were also eliminated if they overlapped other annotated features, such as RNAs, gene features without a CDS, and CDS entries with a '/pseudo' tag, by more than 50%.

For the roughly 350,000 gene predictions that remained, to determine which of these might be true missed genes, we created a BLAST database containing all proteins from every bacterial genome listed in the RefSeq database [29], which was approximately 4.2 million proteins as of March 28, 2011. We then translated and aligned each of the remaining predicted genes against this protein database.

Although the use of homology searches will yield many useful results, some of the false predicted genes may also have a low-scoring alignment that, absent any further processing, would lead us to call this false gene a missed gene. Therefore, to ensure that the pairs of homologous genes implied by these BLAST alignments were indicative of true pairs (as opposed to alignments that could occur simply by random chance), we used three filters: an E-value threshold of 10^{-6}, an alignment coverage requirement, and the requirement that the alignment was to a gene with assigned function. We discuss each in more detail below.

The E-value of an alignment A is the expected number of alignments of the query that would score as well as A by random chance alone, and is dependent on the length of the query and the size of the database searched against [45]. The probability of finding an alignment that scores as well as A (the P-value) is related to the E-value by the equation $P = 1 - \exp(-E)$ [8]. For the E-value 10^{-6}, the P-value is also 10^{-6}, which means the expected number of our 350,000 predicted genes that would align to the database of proteins by chance alone should be less than 1, according to the model used to calculate BLAST E-values. Of the 127,000 possible missed genes that had a BLAST alignment, only 78,000 had an alignment that passed through this filter.

However, many of the proteins in our database are homologous to many others, and so the database is not made of independent sequences. It is unclear as to what degree this would affect the expected number of false homologous pairs. For this reason, as well as the presence of pseudogenes, we further filter our set of alignments by requiring each alignment to cover at least 80% of the subject gene.

This filter reduced our set of possible missed genes to a total of 52,605 candidate missed genes.

Finally, even if a predicted gene were an exact amino acid match to a gene in RefSeq, there is the possibility that the RefSeq gene is simply an incorrect computational prediction. The RefSeq database has many hypothetical proteins, and it is likely that a significant fraction of such genes are not true genes. Those genes that have an assigned function have such an assignment due to either an experiment verifying that function or high sequence similarity to an experimentally-confirmed gene. This means that those genes with assigned function should have a higher probability of being true genes than those without.

Therefore, we divide our remaining candidate missed genes into two groups: those with sequence similarity to a protein without the string "hypothetical" in its description (or any of several common spellings of hypothetical), designated *named missed genes*; and those with sequence similarity only to hypothetical proteins, which we designated *hypothetical missed genes*. Our analysis yielded 13,602 named missed genes and 39,003 hypothetical missed genes that we further analyzed using COMBREX.

2.2.4 COMBREX

COMBREX (Computational Bridges to Experiments) is an NIH funded effort to bring computational and experimental biologists together. It serves as a clearinghouse for computationally determined gene function predictions, prioritizes these for experimental testing, and offers grants to experimental

biologists to test specific predictions [41]. COMBREX maintains a database [42] of experimentally determined and computationally predicted functions for more than 3 million microbial genes. The genes in the COMBREX database are organized into functionally linked gene groups. In the default scheme, the genes are grouped into sequence-similar and likely isofunctional groups, as defined by the NCBI Protein Clusters Database [46].

COMBREX is the first functional database that attempts to provide fully traceable annotation, where predictions are traced (whenever possible) to the experimentally determined evidence. For many genes COMBREX provides a link to the nearest gene with experimentally determined function, as determined by both BLAST similarity score and shared domain composition.

In addition, COMBREX also provides information about documented phenotypes associated with each gene. Currently, this phenotype data consists of antibiotic resistance, antibiotic sensitivity, and candidate gene essentiality. Antibiotic resistance genes, obtained from Antibiotic Resistance Genes Database [47], confer resistance to one or more antibiotics through several mechanisms. Antibiotic sensitivity genes are genes that when lost, confer increased sensitivity to antibiotics [48]. The essential genes are identified, by multiple sources, as being essential for growth or viability in one or more organisms (a complete list of the organisms and sources can be found at the COMBREX web server).

2.2.5 COMBREX Analysis

Approximately 75% of the total missed gene set is comprised of hypothetical missed genes, for which the annotation of related proteins in GenBank provides no useful functional information. However, we were able to provide additional information for many of these hypothetical missed genes with COMBREX, by utilizing the ComBlast annotation pipeline, which we describe below.

Based on the data stored in COMBREX, ComBlast associates each missed gene with various types of important evidence or data. Consider a case of a hypothetical gene prediction identified in two closely related strains using the same gene prediction software. Clearly, this prediction could be a false positive resulting from a roughly similar k-mer distribution in the predicted region. Our confidence in the prediction grows if the same prediction is found in a large number of organisms suggesting this gene is conserved. The confidence also grows if one of the homologs has been explored experimentally. This principle is deployed systematically throughout our study. The evidence associated with each gene includes any one of the following attributes: conservation of the gene, evidence of experimentally validated function or predicted molecular function, existing 3D structures or protein domains, protein purification status, EC numbers, and gene phenotype. A missed gene is associated with specific evidence if it shares significant sequence similarity with individual genes with this type of information, or if it is assigned to one or more protein clusters in COMBREX (with at least one gene in the cluster having the required information). For the purpose of this paper, missed gene A shares significant sequence similarity with

COMBREX gene B if the sequence alignment of A against B covers at least 80% of the length of B and has BLAST E-value less than 10^{-5}. If gene B contains any of the above mentioned evidence, they are assigned to gene A. Gene A can also be assigned to protein cluster C if A shares significant sequence similarity with at least 100 members in that cluster (for big clusters) or all members in the cluster (for smaller clusters), or if A shares significant sequence similarity with members of only one cluster.

The information from COMBREX is used to assign each missed gene to a COMBREX support level. A missed gene is assigned to the *strong* COMBREX support level for being a true functional gene if the gene is found to be conserved in multiple organisms. We define a gene as conserved in multiple organisms if it is assigned to a cluster with more than 50 members coming from at least 2 different phyla or if it shares significant sequence similarity to more than 50 COMBREX genes. Additionally, a missed gene is assigned to the strong COMBREX support level if it has significant similarity to one or more genes associated with at least one of the following types of information: experimentally validated function with evidence, known 3D structure, presence of purified protein, protein domain, or EC number.

A missed gene not assigned to the strong support level is assigned to the *fair* COMBREX support level if it satisfies one or more of the following conditions: if it is assigned to at least one cluster having a computational prediction for its genes, if it is assigned to at least one cluster containing 5 or more members, or if it shares significant sequence similarity to 10-50 COMBREX genes. The

remainder of the named missed genes that do not match the strong or fair criteria, are assigned the *weak* COMBREX support level, since they still have significant sequence similarity to genes with meaningful functional annotation. Other hypothetical missed genes that do not match the strong or fair criteria are labeled as having the *insufficient* COMBREX support level.

2.2.6 Spurious Gene Family Analysis

Along with our use of ComBlast to examine the support available for our candidate missed genes, we used AntiFam [49] to examine our candidate missed genes to determine if there is any evidence that they are not true genes. AntiFam is a database of hidden Markov models (HMMs) that represent families of genes that have been incorrectly annotated in the past frequently. In this study, we use AntiFam 2.0, which contains 47 HMMs representing families that include genes from ORFs that overlap known rRNAs, tRNAs, and other genomic features.

2.3 Results and Discussion

2.3.1 ComBlast Results

By the end of our search for candidate missed genes, we found 13,602 named missed genes with significant sequence similarity to a known gene with non-hypothetical annotation, and 39,003 hypothetical missed genes with homology only to hypothetical proteins. 13,307 of the named missed genes and 36,127 of the hypothetical missed genes had significant sequence similarity to COMBREX genes, and could be further analyzed using COMBREX.

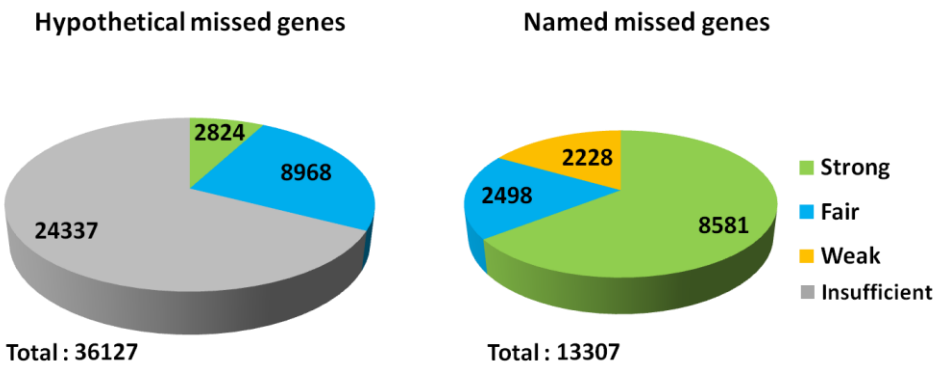

Figure 2. Assignment of COMBREX support levels to the hypothetical/named missed genes using ComBlast. For each missed gene we assign a COMBREX support level based on sequence homology and assignment to gene clusters in COMBREX. A missed gene has the strong COMBREX support level of being a true protein coding gene if it is conserved or associated with at least one of the following information: possessing experimentally validated function, known 3D structure, purified protein, protein domain or EC number. It has the fair COMBREX support level if it has a sufficient number of homologs or is associated with a predicted function. The other named missed genes, which were confirmed by sequence homology to at least one gene with non-hypothetical protein annotation, have a weak COMBREX support level. The rest of the hypothetical genes have insufficient evidence and thus they are not counted in the statistics of missed genes in this paper.

We started by using COMBREX to assign a confidence level to the 13,602 named missed genes. Using ComBlast, the annotation pipeline in COMBREX, we assigned 63% of the named missed genes to the *strong* COMBREX support level and 18% to the *fair* level (Figure 2). In addition, while taking into account the hypothetical missed genes, we were able to double the number of likely genes that have at least the *fair* COMBREX support level for being protein-coding genes. In total, we can assign the *fair* COMBREX support level to another 11,792 genes from the hypothetical missed gene set, which illustrates the limitations associated with sequence homology based prediction methods (Figure 2). 2,824 of the hypothetical missed genes are assigned to the *strong* COMBREX support level (for examples see Table 1), and more than 3,000 genes also have some information through functional predictions of other genes.

Gene	Reasons for association with the strong support level
AE014295_orf00919 from *Bifidobacterium longum* NCC2705	Assigned to the NCBI curated cluster PRK11770. The ORF has 213 significant sequence homologs (BLAST E-values range between 1e-58 to 3e-09) in the cluster. The cluster contains 218 genes from 125 species belonging to 6 different phyla. It has a conserved domain along with few cloned and purified members. Thus, using ComBlast, this hypothetical missed gene is assigned the *strong* COMBREX support level of being a true protein-coding gene.
CP002334_orf00644 from *Helicobacter pylori* Lithuania75	Assigned to the NCBI cluster CLSK496073. All other members of the cluster are hypothetical proteins (NCBI annotation), but COMBREX identified 3 experimentally validated genes within the cluster, giving the *strong* COMBREX support level that this hypothetical missed gene is a true protein-coding gene.
AE017354_orf01466 from *Legionella pneumophila*	Has significant sequence similarity (BLAST E-Value 1e-09) to a gene from *Aeromonas hydrophila* that is included in the gold-standard database in COMBREX (a novel set of genes with experimentally validated molecular function). This gives the *strong* COMBREX support level that the hypothetical missed gene is a true protein-coding gene.
BA000023_orf01717 from *Sulfolobus sokodaii* str. 7	Has significant sequence similarity (BLAST E-Value 2e-21) to a protein from *Sulfolobus solfataricus* with NCBI annotation as hypothetical protein. However, in COMBREX this gene was identified as having a known 3D structure (PDB code: 2JTM) and thus this gene has a higher probability of yielding a functional protein.

Table 1. Examples of hypothetical missed genes that are associated with the *strong* COMBREX support level

We also used COMBREX phenotype data to identify potentially important (scientifically or clinically) missed genes. We could associate 1,264 missed genes with phenotype data stored uniquely in COMBREX (Figure 3A and Table 2). Even genes that could not be associated with any meaningful functional evidence through BLAST-based analysis have been shown to contain interesting phenotype information; we found 46 such cases (see example 3 in the last column of Table 2). In our set of missed genes, we were able to find candidates that could be associated with three phenotypes: 26 could be associated with antibiotic resistance, 210 with antibiotic sensitivity, 852 with candidate essential genes and an additional 176 genes that could be associated both with candidate essential genes and antibiotic sensitivity.

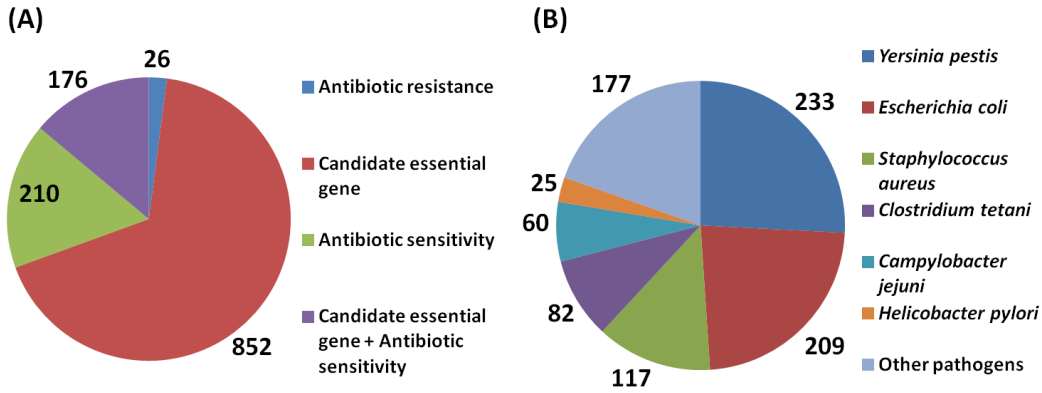

Figure 3. Missed genes that can be associated with COMBREX phenotype data: (A) Phenotype data distribution. Some of the missed genes can be associated with phenotype data using the novel COMBREX resource. A gene is associated with a specific phenotype if it has significant sequence similarity (see Methods) to a gene in COMBREX with the phenotype or if it is assigned to a cluster containing a gene with the phenotype. (B) Potential drug target genes (see text for details) with their distribution in pathogenic organisms (identified only to the species level). The largest portion of these genes belongs to the Enterobacteriaceae family of pathogens (*Yersinia*, *E. coli*, etc.) as seen in the figure. This phenotype information is pooled together using ComBlast.

Along with providing functional and phenotype related information to the missed genes, COMBREX also tries to identify which of these genes might be medically relevant. These genes may be potential drug targets in the future. To identify such cases, we considered 3 different criteria that the missed genes have to satisfy: (i) the missed gene should belong to a pathogenic organism (according to The Microbial Rosetta Stone Database of Pathogens [50]); (ii) the missed gene should be assigned to a protein cluster with at least 50 members (based on ComBlast results); and (iii) the missed gene should possess significant sequence similarity to any essential gene within COMBREX (based on ComBlast results). By this method, we found 359 missed genes, belonging to 88 different pathogenic organisms, that satisfied all the above criteria and thus might be interesting drug targets (Figure 3B). The full list of those genes is also available online [44].

Assigned Name	CP001172_orf00556	CP000948_orf05287	CP002071_orf01199
Species containing missed gene	*Acinetobacter baumannii* AB307-0294	*Escherichia coli* K-12 substr. DH10B	*Helicobacter pylori* Sat464
Species containing homologous gene	*Escherichia coli* ED1a	*Escherichia coli* K-12 substr. MG1655	*Helicobacter pylori* 26695
Name of homologous gene	dimethyladenosine transferase	adenylate cyclase (EC 4.6.1.1)	hypothetical protein
Blast E-value	9e-75	0.0	1e-176
Combrex phenotype associations	antibiotic resistance class KsgA	picin sensitivity and triclosan sensitivity	candidate essential gene

Table 2. Examples of missed genes associated with COMBREX phenotype data. Three representative examples of missed genes that are associated with COMBREX phenotype data: "antibiotic resistance", "antibiotic sensitivity", or "candidate essential gene". Note that CP002071_orf01199 is a hypothetical missed gene with BLAST similarity only to hypothetical proteins.

2.3.2 Spurious Gene Family Analysis

To identify spurious genes in our set of candidate missed genes, the AntiFam database of HMMs was compared against both our named missed genes and hypothetical missed genes. 8 of the 13614 (0.06%) named missed genes, and 141 of the 39003 (0.36%) hypothetical missed genes were labeled as spurious according to AntiFam. We also show the number of spurious missed genes as divided by COMBREX support level in Table 3.

The low number of genes in our named missed genes set that were labeled spurious is encouraging, as we had hoped demanding a functional assignment would result in few false positives. As may be expected, the percentage of genes in the hypothetical missed genes set that are spurious is higher than the named missed genes set, due to the lack of a functional assignment in the homologous genes used to add genes to the hypothetical set.

COMBREX Support Level	Named Missed Genes	Hypothetical Missed Genes
Strong	8 / 8581 (0.09%)	51 / 2824 (1.81%)
Fair	0 / 2498 (0%)	46 / 8968 (0.51%)
Weak	0 / 2228 (0%)	0 / 0 (0%)
Insufficient	0 / 0 (0%)	44 / 24437 (0.18%)
Totals	8 / 13307 (0.06%)	141 / 36127 (0.39%)

Table 3. Spurious genes found within the various subsets of candidate missed genes. The number of spurious genes and number of total genes within each combination of COMBREX support level (strong, fair, weak, and insufficient) and missed gene category (named and hypothetical) is given, as found by running candidate missed genes against the AntiFam database. This table includes only the candidate missed genes that could be analyzed by ComBlast.

Five of the spurious genes (all belonging to *Haemophilus influenzae* F3031) appeared in our named missed genes set due to a single gene annotated in the RefSeq annotation of *H. influenzae* F3031 (HIBPF15861, described as "cell wall-associated hydrolase") that AntiFam indicates is from a region that is antisense to 23S rRNA; a sixth spurious gene was due to a homologous annotated gene in *Lactobacillus crispatus* ST1. The other two spurious genes in this set, called translations of CRISPR regions by AntiFam, are homologs to two genes in *Syntrophus aciditrophicus* SB that were annotated as a "putative cytoplasmic protein" at the time we obtained our set of annotated genes in RefSeq, but now are no longer in the RefSeq record.

We also examined the 51 hypothetical missed genes that were assigned to the strong COMBREX support level, yet were labeled as spurious by AntiFam. One of these genes was a translation of a tRNA, one was contained within a repeat in the *Vibrio* superintegron, and six were contained within the insertion sequence ISlin1. The remaining 43 were labeled as translations of CRISPR regions, with 41 of those being part of a family described by a single HMM. Analysis of these 41 genes revealed that their entry into our candidate missed genes set was, in all but 3 cases, due to homologous genes that existed in RefSeq at the time we

generated the set but that are no longer in the RefSeq database. In addition, all 46 of the spurious hypothetical missed genes assigned to the fair COMBREX support level were translations of antisense rRNA regions.

In summary, many of the spurious genes that we found in our set of candidate missed genes appear to have been introduced into our set by the existence of genes in a slightly outdated version of RefSeq (that have since been removed). There are also many spurious genes that overlapped rRNAs, indicating that the original annotators missed rRNA genes, as our pipeline excluded ORFs that overlapped annotated rRNAs. These two large groups of spurious missed genes, although less than one percent of our total set of missed genes, indicate two other problems with existing annotations that do not involve missing protein-coding genes.

2.3.3 Missed Genes Analysis

In addition to finding missed genes, we conducted further analysis looking for patterns in the data that might explain why these genes were missed. Note that our analysis here is limited to the genes found via our pipeline, and does not attempt to evaluate the annotations beyond analyzing the missed genes we found.

We first checked if the center performing the annotation influenced the number of missed genes in each annotation. The majority of gene annotations were done by 4 major organizations: the Department of Energy Joint Genome Institute (JGI), The Institute for Genomic Research (TIGR), the J. Craig Venter Institute (JCVI), and the Sanger Institute. These four institutes were responsible for over half of all

the annotations that we reviewed in this study. The other centers that provided genome annotations were typically smaller and only did a few genome annotations each.

In analyzing the number of missed genes by each center, we focused solely on analyzing the number of named missed genes, as we had good confidence that most of the named missed genes were true missed genes. When comparing the number of named missed genes from each of the four major centers with the number of named missed genes from the other smaller centers, we found that a higher relative proportion of named missed genes came from smaller centers, suggesting that a lack of adequate resources or experience may have contributed to the higher error rate. Reasons for missing genes might also include using a less sensitive gene finder or gene annotation pipeline.

Overall, for the four major institutes, we found between 0.71 to 3.92 named missed genes for each Mbp of sequence annotated, while for the other smaller centers, the average number of named missed genes found per Mbp was 4.48. We also computed the percent of named missed genes found versus the number of originally annotated genes. Here we found that the four major institutes missed between 0.08% and 0.43% of the genes, while the other centers missed on average 0.48% of the genes. The exact numbers of named missed genes for each of the major centers compared to other centers are provided in Table 4 along with some other statistics.

Center	JGI	TIGR	JCVI	Sanger	Others	Total
Chromosomes annotated	563	95	68	67	781	1574
Named missed genes	1463	852	190	892	10205	13602
Annotated genes	1830805	254484	179284	205667	2105188	4575428
Average missed genes per chromosome	2.60	8.97	2.79	13.31	13.07	8.64
Percent named missed genes vs. annotated genes	0.08%	0.33%	0.11%	0.43%	0.48%	0.30%
Total chromosome length (Mbp)	2058.6	273.9	190.1	227.7	2276.6	5026.9
Named missed genes per Mbp	0.71	3.11	1.00	3.92	4.48	2.71

Table 4. Results of named missed genes analysis. The statistics here only count named missed genes and do not include hypothetical missed genes, as the lack of an association with a known function makes it more difficult to determine if a potential gene is a true gene.

To examine the distribution of missed genes further, we divided the 1,574 annotations into two groups, with one group containing annotations from the four major centers, and the other group containing all other annotations; each group was then ranked by the number of missed genes per Mbp. By plotting each annotation's rank within its group against the annotation's number of missed genes per Mbp (Figure 4), a clear visual distinction between the two groups is evident. A more detailed examination of those annotations with at least 10 missed genes per Mbp reveals that of the 97 such annotations, 79 (81%) were performed by centers other than the four major institutes. Although the major centers do have some annotations that are missing many genes, and some smaller centers miss none or very few genes, clearly the general trend is that the major centers miss fewer genes than the smaller centers. This trend is further confirmed when we examine the distribution of missed gene rates on a per-chromosome basis within the sets of annotations performed by a particular center (Figure 5), as well as view the relationship between the missed gene rate of a center and the number of annotations the center has performed (Figure 6).

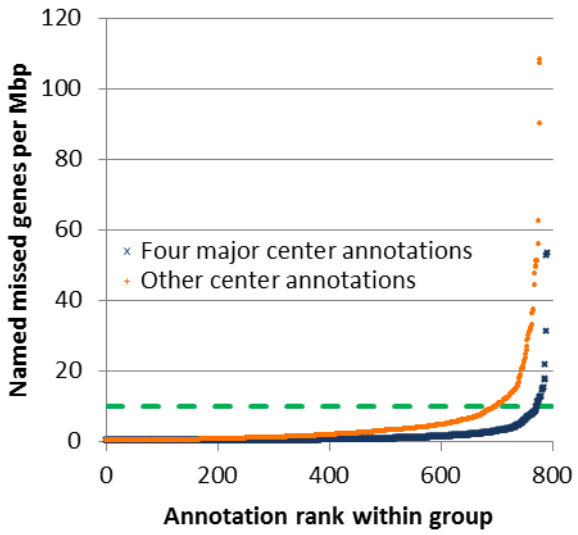

Figure 4. Plot of prokaryotic annotations organized by rate of missed genes. Each of the 1,574 annotations studied was placed into one of two groups: the 793 annotations done by one of the four large centers (JGI, TIGR, JCVI, and the Sanger Institute), and the 781 other annotations done by the smaller centers. The annotations were then sorted for the two groups separately, in increasing order by the number of named missed genes found in the annotation per Mbp of genome sequence. The number of named missed genes per Mbp was then plotted for each annotation according to its rank within the sorted ordering of its group of annotations. The resulting plot shows the distribution of the number of named missed genes per Mbp for annotations provided by the four large centers and by all other centers; each blue 'x' represents an annotation performed by one of the four major annotation centers, while each orange '+' represents an annotation performed by one of the other smaller centers. A green dashed line marks the threshold of 10 named missed genes per Mbp. Note that since there are roughly the same number of annotations performed by the large centers as the smaller centers, the blue 'x's appearing below the orange '+'s indicate that the large centers generally produced annotations with fewer named missed genes per Mbp.

Another observation from our analysis of the set of named missed genes is that many of them are short genes, under 300 bp in length. Examining the shortest annotated gene for each chromosome reveals that some of the annotators likely used a minimum gene length higher than the 110 bp cutoff we used, which may account for some of these short genes being missed. In fact, for the annotations of two strains of *Yersinia pestis*, Z176003 and D182038, all 200+ missed genes were under 300 bp, while the minimum annotated gene length in the original

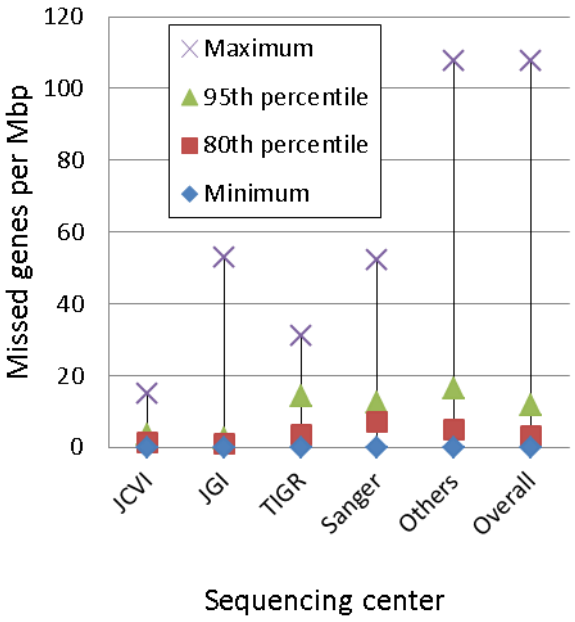

Figure 5. Missed gene rate distributions per center. For each of the four major centers, as well as the other centers as a group, a representation of the distribution of the missed gene rates for the centers is shown. For an individual center, all chromosomes annotated by that center had their missed gene rates calculated, and the 80th and 95th percentiles are displayed along with the minimum and maximum missed gene rates.

annotations was 300 bp. In these cases, it is clear that the minimum gene length setting was the primary cause of these genes being missed.

In our analysis, we used a minimum gene length of 110 bp when running Glimmer3 to find missed genes. This minimum length was used as it is the default length supplied by Glimmer3's g3-iterated.csh script. It appears that based on the smallest annotated gene in each chromosome, some annotators use a higher minimum than 110 bp, which may result in missing shorter genes. Although short genes were not the sole cause of missed genes, we found that about 60% of the named missed genes were genes of length between 110 bp and 300 bp. A full histogram of the lengths of the missed genes we found is shown in Figure 7, and we can see that many of the genes have length less than 300 bp.

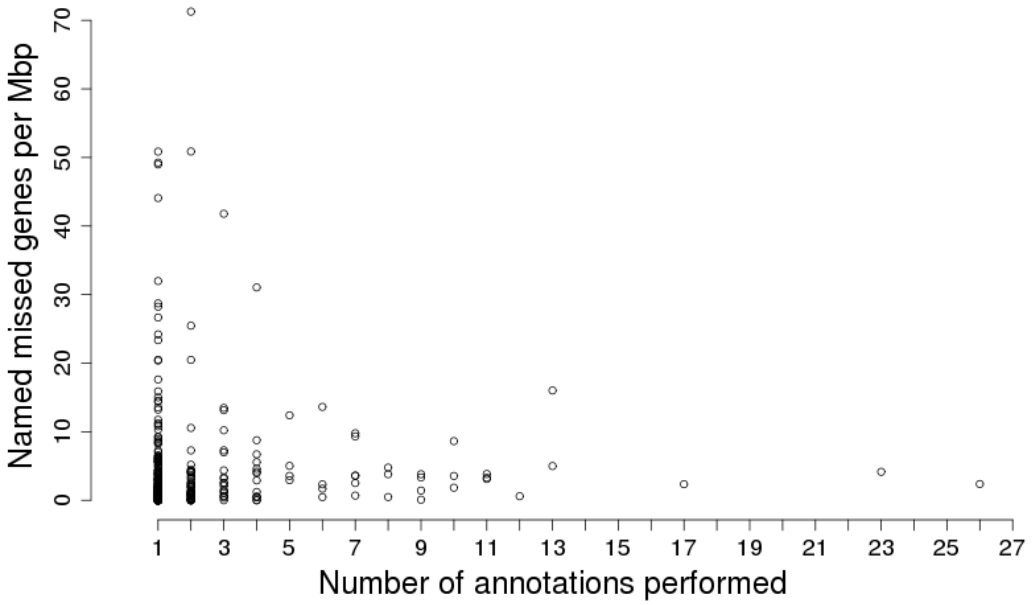

Figure 6. Relationship between the named missed gene rate of a center and the number of annotations performed. Each point on the plot represents a single annotation center, and indicates the number of annotations the center has performed as well as the number of named missed genes per Mbp of sequence annotated by the center. The four major annotation centers (JGI, TIGR, JCVI, and Sanger) do not appear in the figure above, as they each have done over 60 annotations, but they all have less than four named missed genes per Mbp.

In addition to checking the distribution of gene lengths, we also checked to see if the annotations with the highest missed genes were all from several years ago, or if there were still high numbers of genes being missed in recent years. In general, it was difficult to know for certain when each annotation was done, as many annotations did not have publications associated with them, and for the ones with publications, there was still some chance that the annotation was done many years before the publication. However, we did find a number of missed genes in annotations associated with publications from the past five years. There was one particular example where the annotation of *Lactobacillus fermentum* CECT 5716 (accession CP002033) was described in a 2010 publication [51], yet the annotation still had over 100 missing genes per Mbp. This example had only 29

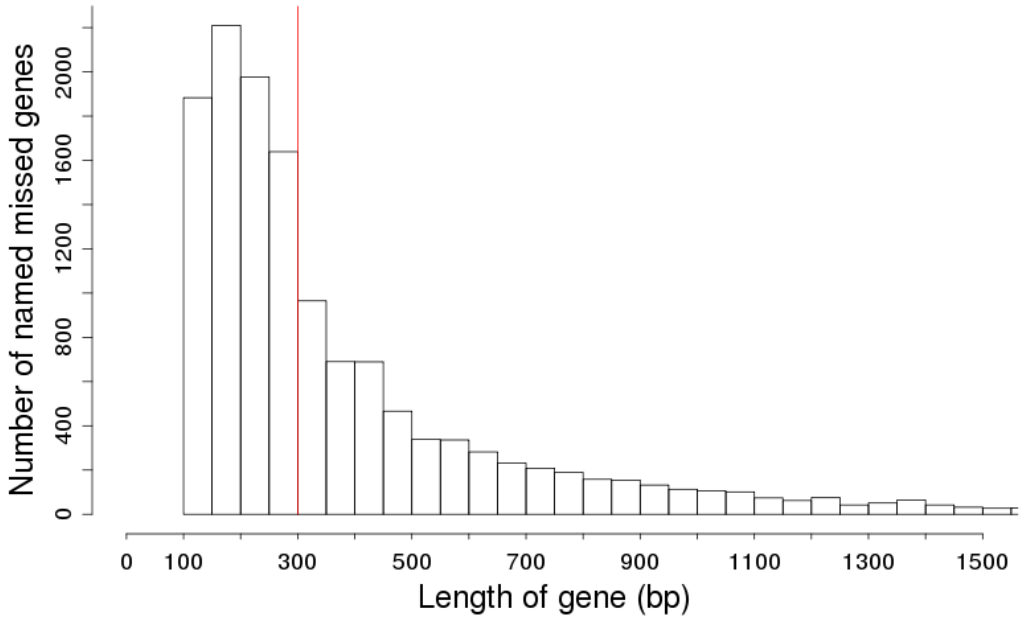

Figure 7. A histogram of the lengths of 13,602 named missed genes found in 1,574 prokaryotic chromosomes. Of the 13,602 named missed genes found in our study, 7,627 were less than 300 bp in length, indicating a strong tendency on the part of some annotators to omit shorter genes from genome annotation.

out of 224 missed genes under 300 bp in length, indicating that recent gene annotations may still miss many genes that are not short. One last factor we checked was to see if the missed gene rate of an annotation was influenced by the GC content of the genome, but we did not see any noticeable correlations (data not shown).

We further examined the ten genomes that had the highest named missed gene rates; these genomes are listed in Table 5. Of these ten, 8 were annotated by smaller centers, with only one annotated by JGI (*Bacillus thurengiensis*) and one by the Sanger Institute (*Streptococcus pneumoniae*). The presence of *E. coli*, *Y. pestis*, and *B. thurengiensis* genomes on this list is especially surprising, given the high number of closely-related genomes that have been sequenced and annotated, which should have made annotating these genomes easier.

Genome	Accession #	Genome Size (Mbp)	Missed genes per Mbp	Total missed genes	% Missed genes ≤ 300 bp	Shortest annotated gene (bp)
N. gonorrhoeae FA 1090	AE004969	2.15	107.7	232	35.3%	108
L. fermentum CECT 5716	CP002033	2.10	106.6	224	12.9%	189
S. glossinidius str. 'morsitans'	AP008232	4.17	89.7	374	25.4%	99
E. coli APEC 01	CP000468	5.08	62.0	315	69.8%	240
C. tetani E88	AE015927	2.80	55.7	156	87.8%	303
B. thurengiensis str. Al Hakam	CP000485	5.26	52.9	278	61.5%	138
S. pneumoniae INV104	FQ312030	2.14	52.3	112	50.0%	99
Y. pestis Z176003	CP001593	4.55	50.9	232	100.0%	300
Y. pestis D106004	CP001585	4.64	50.9	236	99.6%	300
Y. pestis D182038	CP001589	4.63	50.8	235	100.0%	300

Table 5. The ten chromosomes with the highest named missed gene rates.

We were able to find information regarding the gene finding programs used for 7 of these ten annotations; the annotations for *Neisseria gonorrhoeae* [52], *L. fermentum* [51], and *S. pneumoniae* [53] lacked an accompanying publication detailing the annotation methods. Glimmer version 3 was used in the annotation of the three *Y. pestis* genomes [54] in Table 5. Three other annotations (*Sodalis glossinidius*, *Clostridium tetani*, and *B. thurengiensis)* used Glimmer version 2 as part of their annotation [55–57]. In addition, GenomeGambler 1.51 [58] was used to find genes in the annotation for *S. glossinidius* [55], although it is unclear how the results from GenomeGambler and Glimmer were combined. The annotation of *E. coli* used the GeneQuest program sold by DNASTAR for finding genes [59]. Looking at the shortest annotated gene in these ten annotations reveals the likely use of a high minimum gene length in the process of annotation. Five annotations

have no genes less than 240 bp in length, and in each of these five annotations, the majority of missed genes were less than 300 bp in length. This appears to indicate that in these cases, the use of a high minimum gene length was the primary cause of these annotations' high missed gene rates.

In the remaining five cases, the cause of the high rates of missed genes is less clear, largely due to a lack of information about the annotation methods used. For only two of these remaining genomes do we have a description of the gene finders used to perform the annotation, and for *S. glossinidius*, two gene finders' results were used in an ambiguous manner.

We also wanted to determine if the genes missing from the 10 GenBank annotations listed in Table 5 were also missing from their corresponding RefSeq annotations. Although RefSeq does contain manually curated genomes for some organisms, many annotations for bacterial genomes are listed as "provisional", meaning that they have not yet undergone final review by NCBI staff. As many RefSeq annotations for prokaryotic genomes are largely based on the genomes' GenBank annotations when provided [29], the RefSeq annotations may still be missing large numbers of genes. Our comparison between the two sets of 10 annotations, detailed in Table 6, revealed that only 13 genes were unique to RefSeq annotations, while 43 genes were unique to GenBank annotations. Of the 13 genes unique to RefSeq annotations, 11 were in our set of named missed genes.

Genome	GenBank annotation # Genes	# Unique genes	RefSeq annotation # Genes	# Unique genes	Common genes # Genes	# 5' changes
N. gonorrhoeae FA 1090	2002	0	2002	0	2002	9
L. fermentum CECT 5716	1051	0	1051	0	1051	0
S. glossinidius str. 'morsitans'	2432	0	2432	0	2432	13
E. coli APEC 01	4467	39	4430	2	4428	72
C. tetani E88	2373	0	2380	7	2373	21
B. thurengiensis str. Al Hakam	4736	0	4736	0	4736	33
S. pneumoniae INV104	1824	4	1820	0	1820	0
Y. pestis Z176003	3542	0	3546	4	3542	0
Y. pestis D106004	3629	0	3629	0	3629	0
Y. pestis D182038	3620	0	3620	0	3620	0

Table 6. Comparison of GenBank and RefSeq annotations for the ten chromosomes with the highest named missed gene rates. Genes not contained in both annotations (as determined by stop codon position) for a given genome are considered "unique", while those contained in both are "common". For common genes, "# 5' changes" indicates the number of genes that have differing start codon annotations.

2.4 Conclusions

In this study, we found and made publicly available a substantial number of genes missed in the annotations of prokaryotic genomes in GenBank. Through analysis on 1,474 completely sequenced prokaryotic genomes, we found 13,602 genes missed that had significant amino acid sequence similarity to named (non-hypothetical) genes in NCBI's RefSeq database. We also found 39,003 missed genes which had significant sequence similarity only to genes annotated as hypothetical protein, and among those genes, we found 11,792 candidate ORFs with at least some evidence from COMBREX supporting that they are genuine missed genes. The fact that many of the hypothetical genes can be associated with evidence from known genes highlights the need for taking into account more

information than just the gene description as is commonly done by many who use only BLAST. COMBREX addresses this need by providing a wider variety of information coming from many different sources. Given the large number of probable missed genes found by our method, we recommend it as an addition to bacterial annotation pipelines. Although we used Glimmer, we note other high-sensitivity gene finders, or combination of such gene finders, may also be suitable for improving gene annotations.

We found the major centers responsible for genome annotation generally had very few missed genes, and their annotations missed fewer genes on average than annotations from smaller centers. Although some smaller centers did consistently produce gene annotations with low numbers of missed genes, we found that the majority of gene annotations with a high rate of missed genes were from the smaller centers, which may have less experience in gene annotation.

We also found many short missed genes, suggesting that annotators selected a minimum gene length considerably higher than Glimmer's default of 110 bp. Use of such a high minimum length is the likely reason behind the lack of annotation of a large fraction of the missed genes we found. Besides the common problem of missing short genes, it was difficult to determine the reasons for missing the longer genes, as the methods of annotation were not always detailed in GenBank or in the annotations' associated publications. A survey of the genomes that had the 10 highest rates of missed genes showed a variety of annotation pipelines and software, indicating that there are several different approaches to annotation. As these approaches almost certainly differ in terms of sensitivity, those seeking to

perform annotation should take care to ascertain the quality of their chosen methods.

We were also able to identify several genomes without any annotations present, and draft genomes, which appeared on a list of complete genomes on the NCBI Entrez Genome Project website. The presence of such genomes on a list purported to contain "complete genomes" should serve as a reminder to researchers that the data in our public archives is not always 100% accurate. These inconsistencies in genome annotation along with the large number of missed genes found strongly suggests the need for a common standard of best practices to be followed by gene annotation centers, and we hope this work can steer the attention of the annotation community towards this direction.

In addition to identifying many missed genes, we used COMBREX to assign phenotype information to many genes. In our phenotype analysis, we were able associate some of the missed genes with one or more phenotypes, such as antibiotic resistance, antibiotic sensitivity, and essentiality. Some of these missed genes, which are conserved in many organisms and found in potentially pathogenic organisms, could be interesting targets for the pharmaceutical community.

The cost of sequencing is decreasing at a very rapid rate suggesting that the number of organisms or clinically important strains sequenced by small laboratories without bioinformatics expertise will increase dramatically. Our results suggest a need for open access and high accuracy annotation software

available to the community that combines the strengths of gene prediction programs, such as Glimmer, with information from protein databases, such as COMBREX.

3 Improving start site annotation using multiple protein alignments

This chapter describes work with Steven Salzberg that created a pipeline to improve microbial genome annotation by finding regions near the start of genes that were conserved across multiple species. These regions allow the pipeline to determine the start sites of genes with high precision. During our validation of this pipeline, which we call Phantim, we found a number of existing annotations that had a high number of errors with respect to start site annotation. These annotations were brought to our attention by their high disagreement with Phantim's results and a much higher than normal use of rare start codons.

3.1 Background

Many computational methods have been devised for the annotation of prokaryotic genomes. However, the "gold standard" for annotation remains experimental verification of results, a time-consuming task that is undertaken for only a small percentage of genes, a percentage that grows smaller as the number of sequenced genomes continues to rise. As a consequence, the use of automated annotation methods will likely continue to be the dominant approach to genome annotation.

A combination of manually reviewed and automatically generated annotations of genomes is available in NCBI's reference sequence database, RefSeq [28]. Although a portion of the RefSeq database has been manually reviewed by NCBI staff, many genomes' annotations are "provisional," meaning that they have not been reviewed and are usually identical to the annotation submitted by the original genome sequencing group. This means that gene annotations today derive

from many different sources, were created by different methods, and are of varying degrees of quality. Although these issues are widely known, scientists nonetheless rely on published annotation to guide a broad array of downstream research, and most researchers have neither the time nor the expertise to validate the genes they retrieve from public archives. Although perfect accuracy may not be possible with automated method, it is helpful to have some indication as to the quality of a given annotation.

The gene finding programs most widely used for automated annotation typically rely on a combination of statistical models of protein-coding DNA, start site signal detection, and comparative genomics methods. To create models of coding DNA, sets of known or likely protein coding regions are used to train the model; the gene finders Glimmer [2, 3], GeneMark.hmm [37], and Prodigal [5] all work in this fashion. Start sites can be predicted with coding models alone, but their accuracy is rather poor, and a variety of post-processing tools have been created to adjust start site prediction, often focusing on the ribosomal binding site and other signals found upstream of the start codon. Such tools include RBSFinder [31], GS-Finder [60], and TiCO [61]. Recent programs, including Glimmer 3 [4], GeneMarkS [38], and Prodigal [5], have incorporated start site signal recognition into their prediction method to avoid the need for post-processing improvements.

Each of these methods is relatively independent of knowledge of other genomes, allowing them to make predictions on new, unknown genomes. As the number of sequenced genomes has increased, however, useful information is available for annotation of both new and existing genomes. Many of the over 1400 bacterial

genomes in RefSeq are close relatives of each other. Both genome assembly [62] and gene finding [63, 64] methods have exploited the conservation of sequence in these close relatives to improve their results. Close genetic relatives have also been the basis of algorithms to improve start site predictions, using product hidden Markov models [65] and a majority vote approach [66].

In spite of the considerable progress in computational gene prediction, and especially in start site prediction, the accuracy of the methods is difficult to ascertain with certainty. Little experimentally verified data exists for start site locations, and almost all of that data is from one species, *Escherichia coli*. Different gene finders report different accuracies at start site prediction, ranging from 90% to 98.5%, but these estimates are extrapolations that may not be at all accurate for species other than *E. coli* and its close relatives.

The variable nature of most methods' accuracy is due in large part to the fact that they must report a start site prediction for every gene in a genome, even when evidence for a given start site is weak. It is possible instead to report start sites for only a subset of genes, choosing only those for which we can state with high confidence that the start sites are correct. For such genes, comparative evidence can show that the predicted start site is the only one that can explain the sequence conservation seen between genomes. These sets of genes can then be compared to a genome's existing annotation; a high rate of agreement would mean the annotation was highly accurate, while lower agreement could indicate a need to reexamine the annotation.

To this end, we created Phantim (Protein Homology-based ANnoTation IMprovement), a tool utilizing multiple alignments of homologous proteins and a set of strict rules that allow only highly accurate predictions to be made. Phantim makes predictions for only a subset of the genes in a bacterial genome, but our analysis showed these to be 100% correct on a subset of experimentally verified genes, and well over 99% correct on a larger set of genes from 13 species. On several genomes, our analysis indicated not only that existing annotations had mislabeled many start sites, but that they were completely missing some genes, many of which could be found with simple homology searches to a database of known proteins.

3.2 Methods

3.2.1 Summary of the Phantim algorithm

The goal of Phantim is to report a set of genes in a given genome for which there exists evolutionary evidence supporting both the 3' and 5' ends of the reported genes. This is done by comparing each predicted gene in a genome with several homologous genes in closely related genomes, and searching for situations where conservation between the genes implies a necessary protein domain. Where such situations exist, and where they imply an unambiguous start site, Phantim will report the gene as a predicted gene. Through this procedure, Phantim reports a subset of genes for each genome that has high precision with respect to both 3' and 5' ends, and allows for correction of existing annotations.

To begin operation, a user identifies a target genome for Phantim to act upon, and Phantim then examines the set of all genomes to find genomes that are closely

related to the target, which Phantim will use to support its recommendations. The selection of support genomes is performed by first dividing the set of genomes into clusters of genomes that are closely related to each other. Furthest-neighbor clustering is performed with the distances between genomes provided by the Jaccard distances calculated as part of the OperonDB project [67]. Support genomes are then selected such that only one genome per cluster is used.

This selection process is necessary to ensure that the genomes used for comparative purposes in Phantim are not too similar to the target genome. If, for example, two genomes of the same species were used, it is quite possible that alignments between genes from these genomes would show identical stretches of intergenic DNA that are near-identical only because the two genomes have not had time to mutate and drift apart. Our initial attempts at using comparative genomics to improve start site annotation failed for this very reason: the core assumption of Phantim, that conservation will imply functionality, does not hold if the genes being compared are from extremely close relatives. By requiring several more distant relatives to make its decisions, Phantim is able to make much more accurate predictions of genes than it would otherwise.

After support genome selection, Glimmer is used to find possible protein coding genes in both the target as well as all support genomes. BLASTP [8] is then used to find sets of homologous genes among these predicted genes, with several filters used on the results in an attempt to increase the likelihood that support genes used in comparison will be suitable for verifying start site locations. Each set of homologous genes have the genes' protein translations placed into a multiple

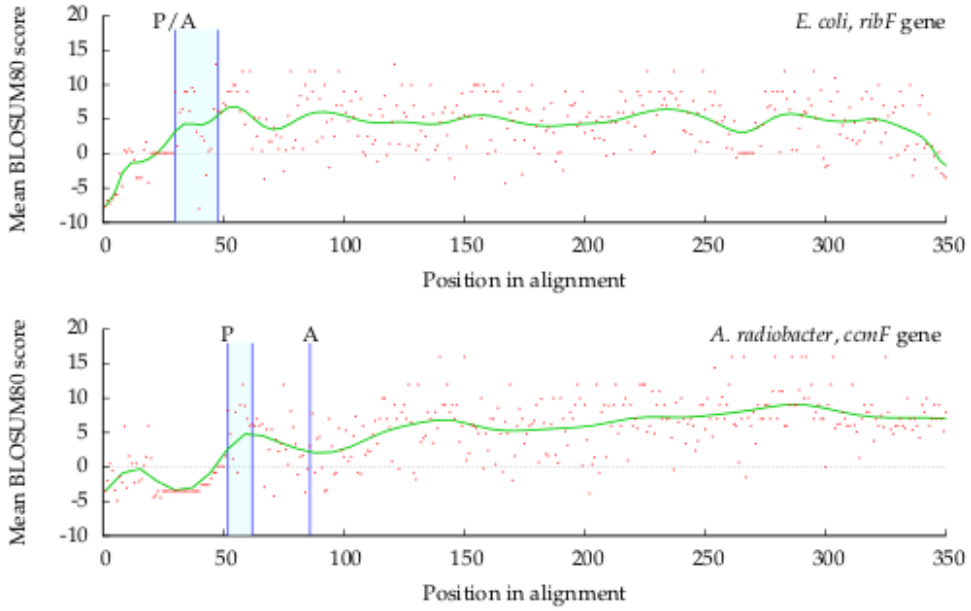

Figure 8. Conservation in the open reading frames of two genes. Conservation is shown in alignments of the *ribF* gene (b0025) in *E. coli* and the *ccmF* gene (Arad_1495) in *A. radiobacter*, each against 10 homologous gene ORFs in different species. The mean BLOSUM80 scores in each column of the alignments are plotted as red dots, with the green line representing the trend of these points. The leftmost blue vertical line in each plot, labeled "P", indicates the first possible start codon, predicted by Phantim. The second blue line is the second possible start codon, and the space between it and the first line is the region examined for conservation. The blue line labeled "A" indicates the position of the annotated start in RefSeq; for *E. coli*'s ribF gene, the prediction and annotation agree, and for *A. radiobacter*'s ccmF gene, the annotation is downstream from the prediction. To simplify the figure, "position in alignment" is relative to the beginning of the target gene's ORF in each alignment, and only 350 positions are shown of each alignment.

alignment using MUSCLE [19], and the alignment is examined for conservation between the first and second possible start codons in the target genome's gene, using a BLOSUM similarity matrix [68] and pairwise comparisons within each column of the alignment. If and only if conservation is found within this region (highlighted in Figures 8 and 9) does Phantim declare the first possible start codon to be the correct one and report the gene as a predicted gene; otherwise, no prediction whatsoever is made regarding the gene.

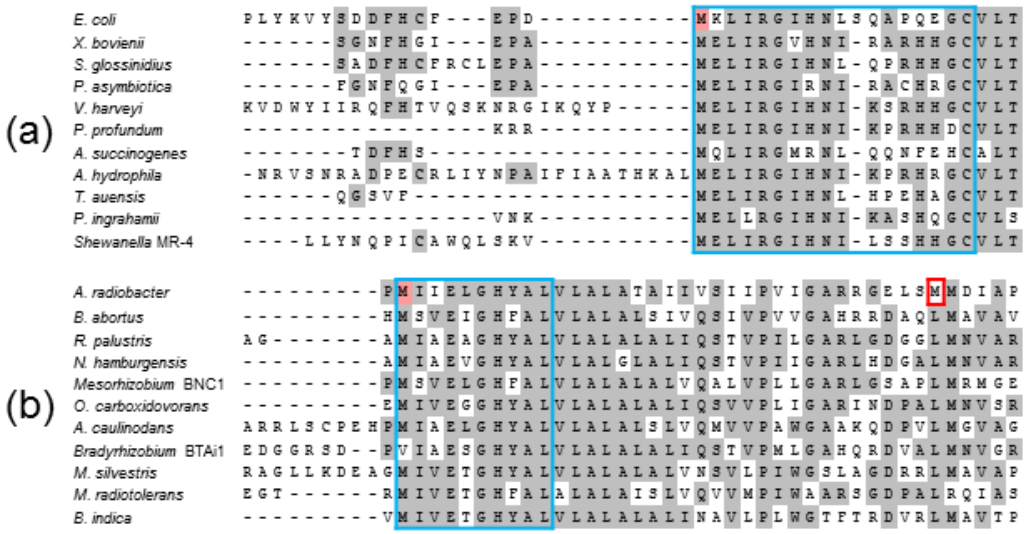

Figure 9. Examples of multiple alignments used by Phantim. Portions of the multiple alignments of ORFs used in Figure 8 are shown, with the examined region between the first two possible start codons in the target gene highlighted with a blue box. The start codon position predicted by Phantim is highlighted with a red background, while the annotated start codon (if different from Phantim's prediction) is highlighted with a red box. (a) The alignment with the *E. coli* gene shows conservation at and immediately downstream of the annotated start site, with a noticeable lack of conservation upstream. (b) The alignment with an *A. radiobacter* gene shows the annotated start site is downstream of the beginning of conservation; the presence of conservation between the first two possible start sites in this gene leads Phantim to declare the first site the true start site. For both alignments, translation is done using the standard translation table, meaning that the GTG codon that serves as an alternate start is translated as valine (V); this is why the leftmost residue inside the blue box in alignment (b) for *Bradyrhizobium* BTAi1 is a V, even though it is actually the annotated start site for that gene and so would be translated as methionine (M) by the ribosome.

For the purposes of Phantim, a region in an alignment is considered to be conserved if the mean BLOSUM80 value meets or exceeds 3.0, and the region is at least 21 nt in length. We determined our threshold scoring value by running Phantim against a set of genes from *E. coli* for which the start site had been verified by N-terminal sequencing. These genes were found by searching the EcoGene database [69] and selecting those genes that were annotated as "Verified" but not as either "EXCEP" or "MUTANT" (in the same manner as was done in evaluating Glimmer3 [4]). Upon trying various scoring thresholds

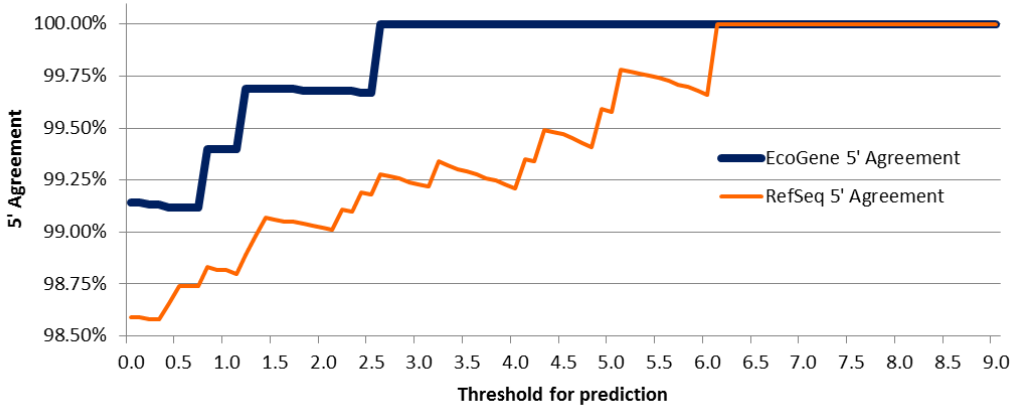

Figure 10. Effect of various scoring thresholds on Phantim's start site prediction agreement. For each half-unit threshold value between 0 and 9 (inclusive), Phantim was run on the *E. coli* K12 substr. MG1655 genome and predictions were only considered for alignments that had an examination region score that was higher than that threshold. The considered start site predictions were then compared to both the RefSeq annotation as well as a subset of genes in the EcoGene database for which the start sites had been experimentally verified. The percentage of genes that had a predicted start site that agreed with the reference annotation (either RefSeq or EcoGene subset) is shown as a function of the various scoring thresholds.

(Figure 10), we settled upon using a threshold of 3.0 to ensure high precision while making as many predictions as possible.

The length requirement was selected to be long enough so that regulatory elements, such as ribosomal binding sites and transcriptional promoters, would not dominate the region and give rise to a false appearance of protein sequence conservation. These two rules, combined with the use of genomes that have a substantial evolutionary distance between them, serve to force Phantim to make only predictions for which it has strong evidence of the start site's correctness.

3.2.2 Selection of support genomes

Phantim begins operation by having a user identify a genome to act upon, called the "target" genome. Then, Phantim must select a set of genomes that: (a) have a close evolutionary relationship with the target genome, and (b) are not *too* closely

related to any other genome in the set (as well as the target genome). To find such a set of what Phantim calls “support” genomes, Phantim utilizes the Jaccard distances between genomes calculated as part of the OperonDB project [67]. The 1059 genomes used in creating OperonDB are clustered using furthest-neighbor clustering, such that no cluster contains two genomes with a Jaccard distance between them that exceeds 0.4 (we have found this to roughly correspond to one species per cluster). This clustering is performed only once, and can be reused between executions of Phantim.

To select the set of support genomes, the following algorithm is used. The set of support genomes, *S*, begins as an empty set, and all 1059 genomes with known Jaccard distances are placed in a set *C* that holds all possible support genome candidates. The target genome, along with all genomes in its cluster, is then removed from *C*. Then the genome *g* in *C* with the smallest Jaccard distance to the target genome is added to *S*; *g*, along with all of the genomes in its cluster, is removed from *C*. This step of selection from and removal from *C* is repeated until either (a) *g* has a Jaccard distance of more than 0.65 from the target genome, or (b) *C* is reduced to the empty set.

3.2.3 Locating coding ORFs and maximal length genes

Once the support genomes are identified, possible genes are found in each genome (including the target) by running Glimmer. A modified version of the g3-iterated.csh script supplied with Glimmer, designed to avoid gene prediction in certain regions, is run against each chromosome and plasmid found in each genome's GenBank record. Phantim reviews the Glimmer predictions for each

chromosome and plasmid, and records three items of information for each gene prediction: (a) the coordinates and amino acid translation of the gene's open reading frame (ORF); (b) the coordinates within the ORF of the first two possible start codons codons within the ORF; and (c) the coordinates and amino acid translation of the maximal length gene (the gene using the first possible start codon as a start codon).

Glimmer was used as the gene finder for Phantim because of its high sensitivity; as only the genes selected in this step are possible candidates for inclusion in Phantim's final report, it is important to have as many true genes found as possible. However, Glimmer's start site predictions are actually ignored by Phantim, as all start site predictions for Phantim will be made using conservation; only Glimmer's 3' and respective ORF predictions are carried forward.

There are certain types of genes that Glimmer does not recognize, however. These include genes with a programmed frame shift, selenoproteins, and RNA genes. Glimmer also cannot determine if a region that looks like a protein coding gene is actually a pseudogene (a region that used to be a gene but is no longer functional). To avoid a situation where Phantim would make predictions that included parts of these kinds of genes (thereby making erroneous predictions), regions that are annotated in GenBank as selenoproteins, pseudogenes, RNA genes, or regions labeled as a gene feature but lacking a corresponding CDS feature are excluded from gene prediction in Glimmer. 50 bp are removed from the edge of each of these excluded regions so that genes that may overlap these regions can still be predicted by Glimmer.

Finally, although Glimmer cannot detect it by itself, if the alternative translation table used by *Mycoplasma* and other related species is specified in the annotation, it will be used by Glimmer as well as all subsequent translations in Phantim.

3.2.4 Finding sets of homologous genes

Phantim requires that several homologous genes in support genomes exhibit conservation with a target gene in order to make a prediction about the target gene's coordinates. To find these homologous genes, Phantim begins by placing all the maximal length genes from the support genomes into a protein database. BLASTP is then run against this database, using the target genome's maximal length genes as query sequences. As the target and support genomes are intended to be closely related, BLASTP is directed to use the BLOSUM80 substitution matrix when scoring its alignments.

The results from BLASTP are filtered in several ways, to increase the likelihood that the support genes used in comparison will be suitable for verifying start site locations. Alignments are discarded if any of the following apply: (a) the alignment's E-value is greater than 1e-3; (b) the identity within the alignment is less than 45%; (c) the alignment's length is less than 90% of the target gene's length; or (d) the difference between the length of the support gene in the alignment and the length of the target gene is more than 10% of the length of the target gene. The list of BLASTP hits is further narrowed by ensuring only one gene from any given support genome can be matched with a given target gene.

The remaining results are then used to create sets of homologous genes, with one set per target gene. Each set consists of the top 10 support genes that have alignments with a given target gene in the filtered BLASTP results. To ensure that predictions are made with sufficient evidence, sets with less than 3 support genes are discarded.

3.2.5 Scoring multiple alignments

For each set of homologous support genes, the genes' respective ORF translations are placed into a multiple alignment along with the ORF translation of the set's corresponding target gene, using MUSCLE [19]. Phantim then examines the window of columns in the multiple alignment that represent the first (inclusive) and second (exclusive) possible start codons in the target gene. Within this window, each column is examined separately, and the amino acids in the support genes are scored by their similarity to the target gene's amino acid within that column, using the BLOSUM80 matrix (gap characters receive a score of -8 for non-identity substitution, and +1 for alignment of two gap characters). For each column, the mean of these scores is calculated, and then the mean of the column scores throughout the window is found. If this mean substitution score is at least 3.0, and the length of the window is at least 7 amino acids, Phantim will declare the first possible start codon to be the correct one and report the gene as a predicted gene; if these two conditions do not hold, no prediction whatsoever is made regarding the gene.

Due to the fact that the similarity between homologous genes can break down toward the 5' ends, a lack of conservation in a given segment of the target gene

does not imply that that segment is not actually part of the gene. In such a case, it is not possible to make a claim one way or another regarding the gene's start site, and so Phantim does not attempt to do so in the absence of conservation. Similarly, only the gap between the first and second possible start codons is searched; conservation here does indeed lend support to the claim that the gap is part of a functional protein. But a lack of conservation in that gap, and the presence of conservation in a later part of the gene does not allow Phantim to make any firm conclusions as to the correct start site, due to the possibility of the target gene possessing a novel mutation not present in the support genes.

3.2.6 Implementation and Execution

Phantim is designed for use with a Linux operating system, and makes extensive use of Perl and Make, along with other standard Unix utilities, in addition to the various programs cited above. To run Phantim with a specific target genome, the name of that genome must be specified. All chromosomes and plasmids in the target genome's GenBank record will have their sequences analyzed for genes that can be annotated with high confidence, and a separate report will be made for each sequence. These reports can then be compared with the GenBank or RefSeq annotation to determine if changes should be made in the current annotation, and all alignments used in making Phantim's predictions are retained for possible manual examination.

As the number of support genomes increases, the amount of gene finding and the size of the BLASTP database increase as well, leading to long running times. To reduce overall execution time, Phantim is designed to use (by default) all

processing units on a computer and store gene finding results for use in future executions.

3.2.7 Evaluation

To evaluate Phantim, we ran it on 15 microbial genomes and compared its predictions to the corresponding annotations in RefSeq; the results of these comparisons are shown in Table 7. We examined manually any predictions that did not match the 5' and 3' ends of the annotated coding sequence, by inspecting the alignment used by Phantim to make the prediction. In the case of predictions without a matching 3' end, we used BLASTP to align the predicted genes and any genes overlapping them against the bacterial component of NCBI's non-redundant protein database.

Our manual examination of the alignments for genes where Phantim's 5' end prediction disagreed with RefSeq's annotation produced one of three recommendations for each gene: *change* to the predicted coordinates, *keep* the existing annotation, or *review* the annotation. For each alignment, sequence conservation was looked for both before and after the predicted start site, as well as before and after the annotated start site. If conservation clearly began at or shortly after the predicted start site, and not at the annotated start site, we recommended that the gene's annotation be changed. In situations where conservation clearly favored the annotated start site, we recommended the annotated be kept as is. In all other situations, where the correct start site is not clear visually (either due to weak conservation or to conservation upstream of the predicted start site), we recommended a review of the annotation. We note that we

did not look for conservation of the start sites themselves, but rather sequence conservation in a region that clearly led to selection of the 5'-most possible start codon in the target gene, either upstream of or at the beginning of sequence conservation.

For each of the genes that were predicted by Phantim but were not present in RefSeq annotations, the gene's source genome was examined along with BLASTP results for the gene. Our recommendations were either to add the gene to annotation, to replace an annotated gene with the predicted one (due to high overlap between the two), or leave the predicted gene out of the set of annotated protein-coding genes while adding it to the annotation as a pseudogene. Given the number of homologs required for a prediction, as well as the high amount of conservation that would be present, we did not expect to declare any of our predicted genes as pseudogenes.

Conclusively evaluating the start-site prediction accuracy of Phantim is difficult due to the lack of experimentally-verified data about start sites. Although a set of genes with experimentally verified start sites exists for *E. coli*, this set was used to set the threshold parameter of Phantim's algorithm, and so we do not use it for evaluation. Instead, we use a set of genes from two archaeal species, *Halobacterium salinarum* and *Natronomonas pharonis*, for which many genes have had their start sites verified through experiment [70]. This set of archaeal genes has the additional benefit of originating from two high-GC genomes, where start-site prediction is considerably more difficult, thereby providing a more exacting test for Phantim than genes from *E. coli*. The set of genes we used

Genome			Gene Counts			Matches with RefSeq Annotation			
Organism	GC	SG	RefSeq	Phantim	%	3’ Matches		5’ & 3’ Matches	
A. citrulli	69	56	4709	471	10	470	99.8%	462	98.1%
A. radiobacter	60	63	6107	564	9	559	99.1%	525	93.1%
B. anthracis	35	19	5328	697	13	696	99.9%	694	99.6%
B. subtilis	44	40	4176	581	14	581	100.0%	577	99.3%
Bradyrhizobium BTAi1	65	48	7393	678	9	677	99.9%	631	93.1%
C. violaceum	65	70	4407	439	10	439	100.0%	424	96.6%
E. coli	51	65	4145	781	19	781	100.0%	775	99.2%
H. pylori	39	9	1573	198	13	198	100.0%	196	99.0%
H. salinarum	68	6	2110	176	8	176	100.0%	175	99.4%
M. tuberculosis	66	16	4189	261	6	258	98.9%	250	95.8%
N. meningitidis	52	28	2063	218	11	218	100.0%	218	100.0%
N. pharonis	63	6	2659	269	10	269	100.0%	269	100.0%
S. aureus	33	31	2650	350	13	350	100.0%	350	100.0%
V. cholerae	47	52	3834	712	19	711	99.9%	711	99.9%
X. bovienii	45	24	4260	848	20	848	100.0%	819	96.6%
Totals			59603	7243	12	7231	99.8%	6632	97.7%

Table 7. Comparison of Phantim predictions to RefSeq annotations. “SG” is the number of support genomes used. 3’ matches are predicted genes that share a stop codon with a CDS in the RefSeq annotation. 5’ & 3’ matches are predicted genes that share both a start and stop codon with an annotated CDS.

excluded plasmid genes, genes without an entry in RefSeq, or genes where the experimentally determined amino acid sequence was ambiguous or disagreed with the RefSeq sequence for the gene.

3.3 Results and Discussion

3.3.1 High agreement in both 3' and 5' predictions

As seen in Table 7, with one exception (*Mycobacterium tuberculosis*), over 99% of Phantim's predicted genes had their 3' ends match with a gene in a genome's respective RefSeq annotation. All genomes also had at least 92% agreement between predictions and 5' end annotations. Genomes with higher GC content tended to have lower 5' agreement, likely a consequence of the increased difficulty in discerning the correct start codon in longer ORFs. In total, of the 7243 genes for which Phantim made predictions, 97.6% agreed with RefSeq on both the start and stop coordinates.

Genome			Gene counts			
Organism	GC%	SG	Verified	Phantim	5’ & 3’ Matches	
H. salinarum	68	6	526	84	84	100.0%
N. pharonis	63	6	321	69	69	100.0%

Table 8. Comparison of Phantim start site predictions to sets of archaeal genes with verified start sites. SG is the number of support genomes used. Gene counts refer only to the genes in the verified subsets; genes predicted by Phantim that lacked a 3’ match within the verified subset were not considered.

This high agreement between our predictions and the RefSeq annotations tends to support the hypothesis that Phantim yields a set of highly accurate predictions. We sought further evidence of Phantim's high precision for start site prediction by comparing the predictions of genes in two archaeal genomes to those with experimentally verified starts (Table 8). Of these 847 genes, Phantim made predictions for 153 (18.1%), and all 153 (100%) agreed with the verified set on both the 5' and 3' ends.

Finally, upon manual examination (Appendices A and B), all of the 3' disagreements resulted in the addition of new unannotated genes. 123 of the 155 5' disagreements were resolved in favor of Phantim, and only 3 were clearly resolved in favor of the RefSeq annotation; each of these 3 were due to the use of a rare ATT or CTG start codon for the gene. Phantim's high percentage of predictions that either agree with the existing annotation, or are verified by examination, gives strong support to the idea that it provides a high-precision set of start-site predictions, and it also provides a rough estimate as to how precise the predictions are. With 99.6% of start-site predictions validated by experimental data or by manual inspection, Phantim appears to be a very precise tool for validating and correcting start-site predictions.

3.3.2 Discovery of previously unannotated genes

Twelve genes predicted by Phantim failed entirely to match an annotated gene from RefSeq. Manual examination of each of these indicated clearly that all twelve of these genes should be added to their respective annotations; e.g., multiple other species contain the same genes, with strong sequence homology. In several cases, these genes should replace annotated genes that have no evidence to support them and whose coordinates overlap. Appendix A contains a summary of the recommended changes and the evidence that exists for making them.

Six of these genes exist in regions that are currently annotated as intergenic in RefSeq, or are overlapped by less than 5 nt by another gene. A simpler method of extracting these regions, supplying them as input to TBLASTN, and searching against a database of known bacterial proteins would have found as much evidence as Phantim did for these genes, if not far more. Such a method would also find such evidence much faster, and almost certainly would find even more unannotated genes than Phantim. Given the large number of bacterial genomes that have been sequenced and annotated, such a process should now be used by any annotation pipeline.

Six other predicted genes have large overlaps with annotated genes, in most cases being completely overlapped by the annotated gene. In each of these six cases, the gene predicted by Phantim has far more alignments against bacterial proteins in NCBI's non-redundant database than does the corresponding overlapping annotated gene. This larger amount of sequence conservation in other bacteria found in the predicted genes serves as strong evidence of the correctness of these

predictions; in light of the high overlap in these situations, the higher conservation also serves as evidence of the incorrectness of the conflicting annotations.

Like those genes found in regions annotated as intergenic, the predicted genes that were overlapped were all found as part of the Glimmer prediction set. Even in the face of a conflicting annotation from another source, the conflict can be resolved by the use of a BLAST search against other genomes. In this case, a TBLASTN search of two overlapping potential genes against all bacterial genomes provides guidance as to which of the two genes was more conserved, and thus more likely to be a true gene. Once again, with the rise in sequenced bacterial genomes in GenBank, there exist considerable resources for such a method, and it would be a useful addition to an annotation pipeline.

3.3.3 Lengthening predicted genes

The primary motivation behind Phantim is to correct start site annotations, and it appears to perform this task with very high precision. For all but one genome, over 99% of Phantim's start site predictions either matched the RefSeq annotation or were verified by examination. When Phantim's start site prediction disagreed with the RefSeq annotation, manual review of the alignments favored Phantim in nearly all cases. These results are summarized in Table 9, with details in Appendix B. This high percentage of gene predictions that agree with the annotation or have been verified supports the idea that Phantim provides a high-precision set of start site predictions, and it also provides a rough estimate as to exactly how precise the predictions are.

Genome		Phantim comparison		Recommendations				
Organism	GC	3’ matches	5’ mismatches	Chng.	Rev.	Keep	5’ matched/verified	
A. citrulli	69	470	8	6	2	0	468	99.6%
A. radiobacter	60	559	34	24	10	0	549	98.2%
B. anthracis	35	696	2	2	0	0	696	100.0%
B. subtilis	44	581	4	3	0	1	580	99.8%
Bradyrhiz. BTAi1	65	677	46	42	4	0	673	99.4%
C. violaceum	65	439	15	11	4	0	435	99.1%
E. coli	51	781	6	3	1	2	778	99.6%
H. pylori	39	198	2	2	0	0	198	100.0%
H. salinarum	68	176	1	1	0	0	176	100.0%
M. tuberculosis	66	258	8	7	1	0	257	99.6%
N. meningitidis	52	218	0	0	0	0	218	100.0%
N. pharonis	63	269	0	0	0	0	269	100.0%
S. aureus	33	350	0	0	0	0	350	100.0%
V. cholerae	47	711	0	0	0	0	711	100.0%
X. bovienii	45	848	29	22	7	0	841	99.2%
Totals		7231	155	123	29	3	7231	99.6%

Table 9. Results of examination of 5’ differences in prediction. All Phantim predictions were compared to the respective RefSeq annotation, and for those genes that had a 3’ match but a 5’ mismatch, the alignment was analyzed. Recommendations as a result of that analysis are given here, with “Chng.” meaning an annotation should be changed, “Rev.” indicating further review is needed, and “Keep” meaning that the annotation should be left as is. “5’ matched/verified” is a count of predicted genes with 3’ matches and a 5’ end that either matched a RefSeq gene or was verified by examination, along with the percentage of 3’ matches such a count constitutes.

Because Phantim will only predict a start site that lies at the 5'-most start codon, the gene predicted by Phantim is always at least as long as the corresponding annotated gene (with the exception of genes that use very rare start codons). In some cases, Phantim extends genes by more than 100 nt in the 5' direction. With the exception of *Xenorhabdus bovienii*, in most of the low-GC (< 60%) genomes, the number of genes needing adjustments was very low. Many more start sites needed to be altered in high-GC genomes, which are more prone to misannotation due to the longer ORFs found out-of-frame in such genomes.

3.3.4 High erroneous annotation of rare start codons

There appears to be a connection between the quality of an annotation and the number of rare start codons used in the annotation. Most bacterial gene finders

will only allow ATG, GTG, and TTG start codons with their default settings, although the NCBI standard for bacterial annotation also permits translation initiation with CTG, ATT, ATC, and ATA. These rare start codons have been shown to be used in exceptional cases; e.g., when an organism needs to limit the production of IF3 [71, 72]. For the two most intensively studied species, *Bacillus subtilis* and *E. coli*, only 21 out of 8321 protein-coding genes (0.25%) use one of these four rare start codons. For many other genomes, no instances of rare start codons appear in their RefSeq annotations.

While other species might use a significantly higher proportion of rare start codons, computational prediction methods cannot justify their use without significant evidence. Looking at the annotation for the 13 genomes used in Phantim's evaluation, four had > 0.7% of genes utilizing the CTG codon. (None of these used any of the other rare starts.) These four genomes also had the four largest numbers of disagreements with Phantim for predicted start sites when compared to the other genomes in this study (Table 10, last two columns). The only common property among these genomes appears to be their higher-than-expected proportion of CTG start codons. Curiously, there does not appear to be any documented reason for such a high usage of this rare start codon for any of these species [73–75].

The high rate of Phantim's disagreement over a small subset of genes in these species appears to indicate a weakness in the methods used to annotate their start sites. In particular, by allowing their annotation methods to automatically call a CTG start codon, the authors of these studies might have unintentionally over-

Genome		RefSeq start codon usage (%)				Phantim predictions	
Organism	GC%	CTG	ATT	ATC	ATA	5’ mismatches	Changes
A. citrulli	69	0.00	0.00	0.00	0.00	8	6
A. radiobacter	60	3.95	0.00	0.00	0.00	34	24
B. anthracis	35	0.00	0.00	0.00	0.00	2	2
B. subtilis	44	0.14	0.22	0.05	0.00	4	3
Bradyrhizo. BTAi1	65	0.72	0.00	0.00	0.00	46	42
C. violaceum	65	4.29	0.00	0.00	0.00	15	11
E. coli	51	0.05	0.05	0.00	0.00	6	3
H. pylori	39	0.06	0.06	0.00	0.00	2	2
H. salinarum	68	0.00	0.00	0.00	0.00	1	1
M. tuberculosis	66	0.00	0.00	0.00	0.00	8	7
N. meningitidis	52	0.00	0.00	0.00	0.00	0	0
N. pharonis	63	0.00	0.00	0.00	0.00	0	0
S. aureus	33	0.00	0.00	0.00	0.00	0	0
V. cholerae	47	0.00	0.00	0.00	0.00	0	0
X. bovienii	45	3.92	0.00	0.00	0.00	29	22

Table 10. Relationship between GC content, high annotation of rare start codons, and Phantim disagreement. RefSeq start codon usage is the percentage of annotated genes using the given start codon. 5’ mismatches are instances where the prediction had a stop codon match with the annotation, but not the start codon. Changes are the number of 5’ mismatches resolved in favor of the prediction after manual examination.

predicted the usage of CTG starts. For this and other rare start codons, we would suggest that any software that predicts rare starts should only do so when there is considerable evidence (ideally, experimental evidence) supporting those predictions. Otherwise, a high rate of annotation of CTG or other rare start codons may simply indicate that the existing annotation is in need of review.

An inspection of the bacterial chromosome annotations in RefSeq revealed five genomes that used rare start codons for more than 5% of genes, and had at least 50 such genes. Three of these genomes, of the *Mycoplasma* genus, were annotated by the same group. Four of the five annotated ATT as the start site for 13-20% of their genes. An examination of the articles published for these genomes revealed no discussion of these rare start codons [76–78]. We ran Phantim on each, and found extraordinarily high disagreement between the annotations and Phantim's 5' predictions (see Tables 11 and 12). This level of disagreement, combined with

Genome		RefSeq start codon usage (%)			
Organism	GC%	CTG	ATT	ATC	ATA
C. turicensis z3032	57	8.36	5.06	4.51	3.06
M. hyopneumoniae 7448	28	1.37	17.81	7.91	3.65
M. hyopneumoniae J	29	1.37	17.96	8.68	3.96
M. synoviae 53	29	0.46	13.51	9.71	2.28
R. massilae MTU5	33	2.58	19.94	23.14	4.65

Table 11. Five genomes with high annotated usage of rare start codons. Start codon usage is a percentage of all genes annotated for a given genome.

Genome		Gene counts		Matches with RefSeq annotation			
Organism	SG	RefSeq	Phantim	3' Matches		5' & 3' Matches	
C. turicensis z3032	54	4213	934	932	99.8%	768	82.4%
M. hyopneumoniae 7448	7	657	85	85	100.0%	71	83.5%
M. hyopneumoniae J	7	657	85	85	100.0%	71	83.5%
M. synoviae 53	10	659	109	109	100.0%	86	78.9%
R. massilae MTU5	7	968	80	80	100.0%	38	47.5%

Table 12. Results of running Phantim on five genomes with high annotated rare start codon usage.

Phantim's high precision and a startlingly high rate of rare start codons, leads us to suggest that these five genomes contain many incorrectly annotated start sites.

3.3.5 Factors affecting the number of Phantim's predictions

High-GC genomes have ORFs that extend farther upstream from the true start codon than do low-GC genomes. This property is a simple consequence of the fact that stop codons (TAA, TGA, TAG) are AT-rich, and in GC-rich genomes, fewer of these triplets are found by chance in intergenic regions. As shown in Table 7, Phantim predicts fewer start sites in high-GC genomes than it does in low-GC genomes. This is due to two factors associated with a longer upstream region. First, the extra possible start codons can cause the maximal length genes to be of widely differing sizes in different genomes; Phantim's requirement for homologous maximal length genes to be of approximately the same length can eliminate possible useful homologs from consideration. This in turn can lead to a target gene not having enough homologs in other species, which will lower the

number of predicted genes. Another factor that drives the prediction count down in high-GC genomes is that Phantim will only predict genes where it finds evidence that the first possible start codon is the correct one; the more possible start codons upstream of true start codons that a genome has, the fewer genes that can be predicted by such a method. This requirement might be relaxed for future versions of Phantim.

In addition, low numbers of support genomes will result in very few predictions. This occurs when an organism has not had many of its closer evolutionary neighbors sequenced. As the bacterial tree of life is filled out due to future sequencing efforts, Phantim will be able to predict many more start sites.

3.4 Conclusion

Phantim's gene predictions are highly accurate, with a precision that exceeds 99%. Such high precision enables it to make corrections to existing annotations. Application of Phantim to existing genomes reveals that species with a high rate of rare start codons are likely to have a high error rate in start site annotation. In addition to recommending more accurate start sites, Phantim also discovered a small number of genes that were entirely missing from the current annotation. Approximately half of these would have been discovered by a thorough BLAST search of the annotated intergenic regions.

The results of comparing Phantim's predictions with RefSeq's annotations reveal that researchers seeking to use these annotations should be aware that although these annotations are mostly correct, they may contain errors, particularly in the

location of gene start sites. Researchers should especially cautious of annotations that use rare start codons. At the same time, genome annotators should ensure that their software pipelines generate results that are consistent with known genomes and with experimental evidence.

4 Ultrafast metagenomic sequence classification using exact alignments

This chapter describes work performed with Steven Salzberg to provide a highly sensitive metagenomic sequence classification program that performs its task in a fraction of the time used by existing methods. This program, called Kraken, uses exact alignments of short DNA sequences (k-mers) to achieve accuracy comparable to classification methods that use more computationally expensive methods such as inexact alignments or Markov models. Using these exact alignments, along with a pre-computed database that maps k-mers to nodes in a taxonomic tree, Kraken is able to achieve this accuracy nearly a thousand times faster than the fastest comparable classification methods. This work was published in *Genome Biology* in March 2014 [23].

4.1 Background

Metagenomics, the study of genomic sequences obtained directly from an environment, has become an increasingly popular field of study in the past decade. In projects that have studied environments as varied as sea water [79], acidic mine drainage [25], and the human body [32], metagenomics has allowed researchers to create a picture of an environment's microbial life without the need to isolate and culture individual microbes. Combined with the ability to quickly sequence DNA, metagenomics projects can generate a huge amount of sequence data that describes these previously invisible worlds.

For many metagenomic samples, the species, genera, and even phyla present in the sample are largely unknown at the time of sequencing, and the goal of

sequencing is to determine this microbial composition as precisely as possible. Of course, if an organism is completely unlike anything previously seen, then its DNA sequence cannot be characterized other than to label it as novel. Many species, though, have some detectable similarity to a previously known species, and this similarity can be detected by a sensitive alignment algorithm. The most well-known such algorithm, and one of the best methods for assigning a taxonomic label to an unknown sequence, is the BLAST program [8], which can classify a sequence by finding the best alignment to a large database of genomic sequences. Although BLAST was not designed for metagenomic sequences, it is easily adapted to this problem and it remains one of the best methods available [22].

Other methods of sequence classification have been proposed, utilizing sequence alignment and machine learning techniques in an attempt to improve upon BLAST's accuracy. In the MEGAN [80] program, a sequence is searched (using BLAST) against multiple databases, and the lowest common ancestor (LCA) of the best matches against each database is assigned to the sequence. PhymmBL [22, 81] combines the results of BLAST with scores produced from interpolated Markov models (IMMs) to achieve higher accuracy than BLAST alone, and the Naïve Bayes Classifier (NBC) [82] applies a Bayesian rule to distributions of k-mers within a genome. However, all these programs perform at speeds slower than BLAST, which itself takes very substantial CPU time to align the millions of sequences generated by a typical Illumina sequencing run. This processing

burden is so demanding that it suggested another, faster approach to metagenomic sequence analysis: abundance estimation.

Abundance estimation programs work by creating a database that is much smaller than the collection of all genomes, which allows them to perform classification much faster than methods that attempt to identify every read in a data set. These databases are engineered to contain "marker" genes (single-copy genes present in nearly all microbes) [34], or genes that have been found to be specific to certain clades [33]. Because the databases only contain a very small sample of each genome, these programs can only classify a small percentage of sequences from a typical metagenomics sample. They are meant to be used to characterize the distribution of organisms present a given sample, rather than labelling every single read. For example, the initial analysis of the Human Microbiome Project [32] used one of these programs, MetaPhlAn [33], to perform an analysis of several trillion bases (terabases) of metagenomic sequence collected from hundreds of humans. Although abundance estimation programs provide a summary-level characterization of a metagenomics project, they cannot help with analyses that require more details about the sample. For example, they cannot be used to estimate the gene content in a sample, which requires every read to be compared to known genes. If a sample contains a large number of reads from one species, then it is sometimes possible to assemble those reads to reconstruct part or all of the genome [83], and then to classify the resulting contigs.

Here we describe Kraken, a new sequence classification tool whose accuracy is comparable to the best existing sequence classification techniques, and whose

speed far exceeds both classifiers and abundance estimation programs. This speed advantage derives in large part from the use of exact-match database queries of k-mers, rather than inexact alignment of sequences. Its accuracy is made possible by the very large and still-growing number of sequenced microbial genomes, currently numbering over 8,500, which makes it likely that very similar sequences from a given species have been seen before. Through the use of a novel algorithm to process the disparate results returned by its database, Kraken is able to achieve genus-level sensitivity and precision that are very similar to that obtained by the fastest BLAST program, Megablast.

4.2 Results and Discussion

4.2.1 K-mer to LCA database

At the core of Kraken is a database that contains records consisting of a k-mer and the lowest common ancestor (LCA) of all organisms whose genomes contain that k-mer. This database, built using a user-specified library of genomes, allows quick lookup of the most specific node in the taxonomic tree that is associated with a given k-mer. Sequences are classified by querying the database for each k-mer in a sequence, and then using the resulting set of LCA taxa to determine an appropriate label for the sequence (Figure 11 and Methods). Sequences that have no k-mers present in the database are left unclassified by Kraken. By default, Kraken builds the database with $k = 31$, but this value is user-modifiable.

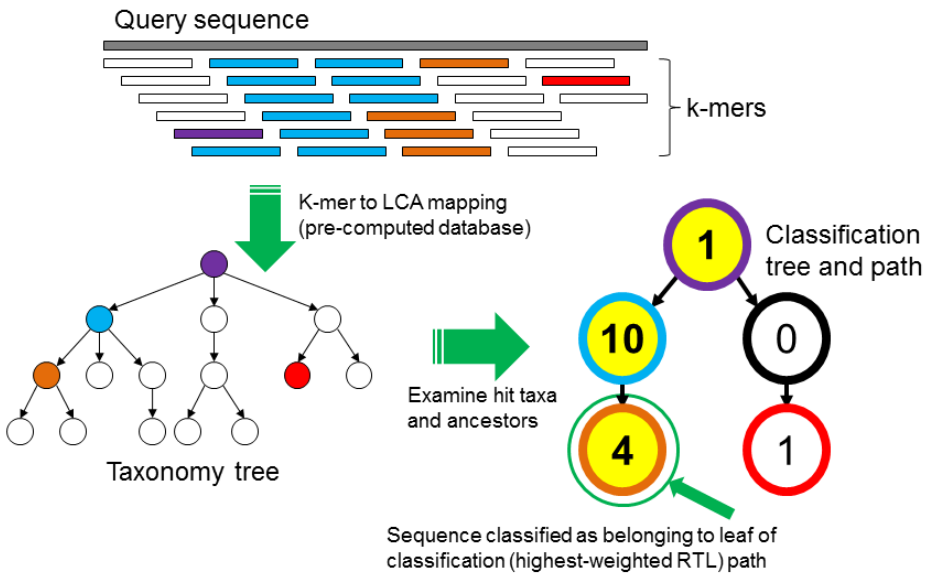

Figure 11. The Kraken sequence classification algorithm. To classify a sequence, each k-mer in the sequence is mapped to the lowest common ancestor (LCA) of the genomes that contain that k-mer through the use of a database. The taxa associated with the sequence's k-mers, as well as the taxa's ancestors, form a pruned subtree of the general taxonomy tree that is used for classification. In the classification tree, each node has a weight equal to the number of k-mers in the sequence associated with the node's taxon. Each root-to-leaf (RTL) path in the classification tree is scored by adding all weights in the path, and the maximal RTL path in the classification tree is the classification path (nodes highlighted in yellow); the leaf of this classification path (the orange, leftmost leaf in the classification tree) is the classification used for the query sequence.

4.2.2 Simulated metagenome data

Although genuine metagenomic reads might provide the most realistic test of performance, such data would not allow us to assess classification accuracy, because the true species in metagenomic data sets today are mostly unknown. We instead used two simulated metagenomes created by combining real sequences obtained from projects that sequenced isolated microbial genomes. When creating these simulated metagenomes, we used data sequenced by the Illumina HiSeq and MiSeq sequencing platforms, and thus we call these the "HiSeq" and "MiSeq" metagenomes respectively (see Methods). These metagenomes were constructed to measure classification speed and genus-level accuracy for data generated by current and widely-used sequencing platforms.

In addition to the two simulated metagenomes constructed with sequences from isolated genomes, we created a third metagenomic sample covering a much broader range of the sequenced phylogeny. This sample, featuring simulated bacterial and archaeal reads (called simBA-5), was created with an error rate 5 times higher than what would be expected, in order to evaluate Kraken's performance on data that contain many errors or have strong differences from Kraken's genomic library (see Methods).

4.2.3 Classification accuracy

Classifiers generally adopt one of two strategies: for example, PhymmBL and NBC classify all sequences as accurately as possible, while Kraken and Megablast leave some sequences unclassified if insufficient evidence exists. Because PhymmBL and NBC label everything, they will tend to have more false positives than methods like Kraken. In turn, one can expect a selective classifier to have higher precision at some cost to sensitivity. Uniquely among metagenomics classifiers, PhymmBL supplies confidence scores for its classifications, which can be used to discard low-confidence predictions and improve accuracy. Using a lower bound of 0.65 for genus-level confidence, we created a selective classifier based on PhymmBL's predictions that we denote as PhymmBL65. To compare Kraken's accuracy to these other classification methods, we classified 10,000 sequences from each of our simulated metagenomes and measured genus-level sensitivity and precision (Table 13 and Figure 12). Here, sensitivity refers to the proportion of sequences assigned to the correct genus. Precision, also known as positive predictive value, refers to the proportion of correct classifications, out of

	HiSeq		MiSeq		simBA-5	
Classifier	Prec	Sens	Prec	Sens	Prec	Sens
Megablast	99.03	79.00	92.44	75.76	96.93	93.67
NBC	82.33	82.33	77.78	77.78	97.64	97.64
PhymmBL	79.14	79.14	76.21	76.21	96.11	96.11
PhymmBL65	99.13	73.95	92.47	73.03	99.08	95.45
Kraken	99.20	77.15	94.71	73.46	99.90	91.25
Kraken-Q	99.12	76.31	94.69	70.41	99.92	89.54
MiniKraken	99.44	66.12	97.41	67.95	99.95	65.87
MiniKraken-Q	99.36	65.67	97.32	65.84	99.98	65.31
Kraken-GB	99.51	93.75	98.48	86.23	99.48	91.13

Table 13. Genus-level classification accuracy against three simulated metagenomes. Precision and sensitivity, measured at the genus rank, are displayed for each of the classifiers used.

the total number of classifications attempted. Kraken's sensitivity and precision are very close to that of Megablast; for all three metagenomes, Kraken's sensitivity was within 2.5 percentage points of Megablast's. The use of exact 31-base matches, however, appears to yield a higher precision for Kraken, as its precision was the highest of all classifiers for each of the three metagenomes. As may be expected, the nonselective classifiers were able to achieve slightly higher sensitivity than the selective classifiers, but at the cost of a significantly lower precision, approximately 80% versus close to 100% for Kraken.

We also note the recent publication of a method, LMAT [84], which uses a k-mer indexing scheme similar to Kraken's, but otherwise differs in its classification strategy. LMAT cannot easily be downloaded and run on our simulated data (see Appendix C) so instead we ran Kraken on a data set used in LMAT's published results. On that data (the "PhymmBL" set), Kraken exceeded LMAT's accuracy results at both the tasks of identifying read origin and identifying the presence of species in the sample. Both methods had essentially perfect (near 100%) precision, but Kraken correctly labelled the species of 89% of the reads while LMAT only did so for 74% of the reads. However, as we note, that data set does

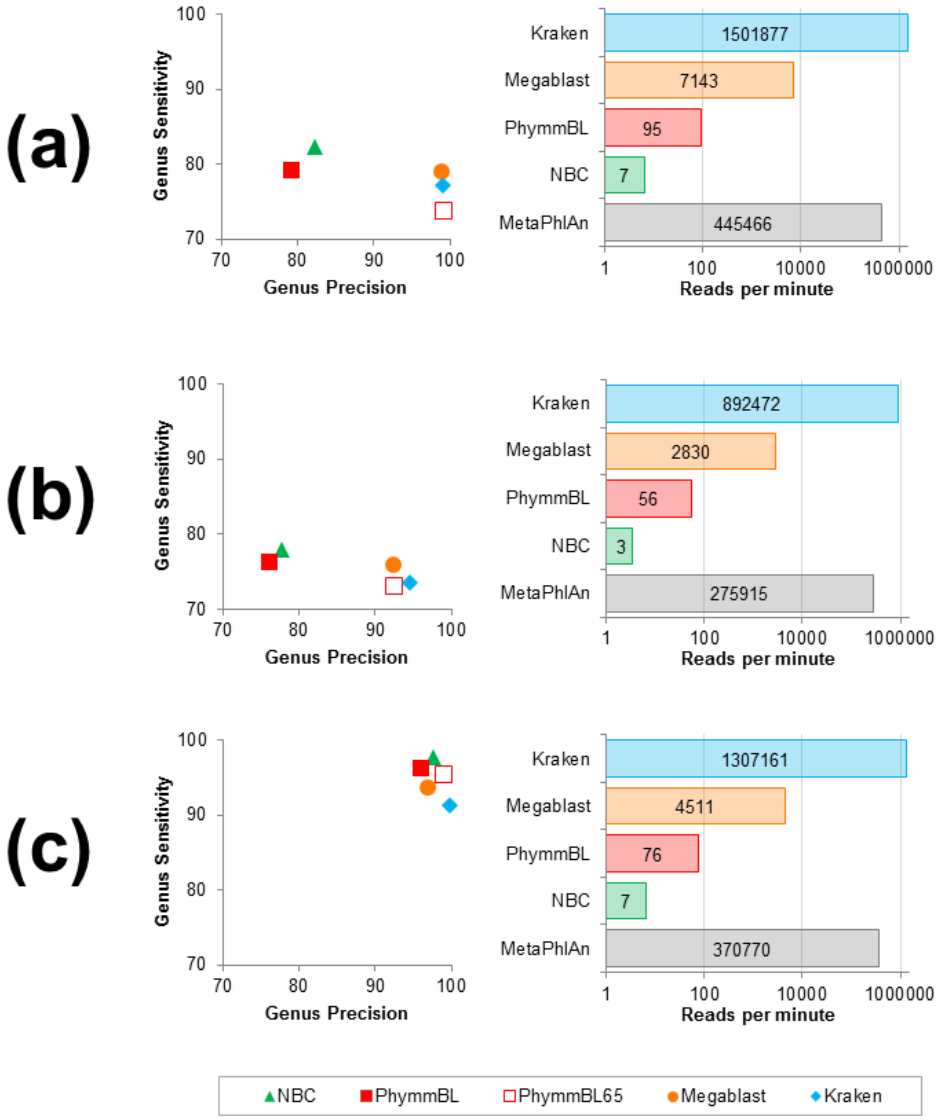

Figure 12. Classification accuracy and speed comparison of classification programs across three simulated metagenomes. For each metagenome, genus precision and sensitivity are shown for five classifiers, and speed is shown for five programs (PhymmBL65 is simply a confidence-filtered version of PhymmBL's results, and MetaPhlAn only classifies a subset of reads that map to one of its marker genes, as it is an abundance estimation program). Results shown are from performance on (a) the HiSeq metagenome, consisting of HiSeq reads (mean length μ = 92 bp) in equal proportion from 10 bacterial sequencing projects; (b) the MiSeq metagenome, consisting of MiSeq reads (μ = 156 bp) in equal proportion from 10 bacterial projects; and (c) the simBA-5 metagenome, consisting of simulated 100 bp reads with high error from 1,967 bacterial and archaeal taxa. Note that the horizontal axes in all speed graphs are shown with a logarithmic scale.

not provide a good basis for comparison because the reads are simulated without error from genomes included in both Kraken's and LMAT's databases.

4.2.4 Classification speed

Because of the very large size of metagenomic data sets today, classification speed is critically important, as demonstrated by the emergence of rapid abundance estimation programs such as MetaPhlAn. To evaluate classification speed, we ran each classifier, as well as MetaPhlAn, against each of our three metagenomes that we used to test accuracy (Figure 12).

Kraken classified reads much faster than any other classifier, with performance ranging from 150-240 times faster than the closest competitor. Kraken processed data at a rate of over 1.5 million reads per minute (rpm) for the HiSeq metagenome, over 1.3 million rpm for the simBA-5 metagenome, and over 890,000 rpm for the MiSeq metagenome. The next fastest classifier, Megablast, had speeds of 7,143 rpm for the HiSeq metagenome, 4,511 rpm for the simBA-5 metagenome, and 2,830 rpm for the MiSeq metagenome. For all three metagenomes, PhymmBL classified at a rate of <100 rpm, and NBC at <10 rpm. Kraken is also more than three times as fast as MetaPhlAn (which only classifies a subset of reads), which had speeds of 445,000 rpm; 371,000 rpm; and 276,000 rpm for the HiSeq, simBA-5, and MiSeq metagenomes, respectively. These results are shown in Figure 2. As expected, all tools processed the longer MiSeq reads (mean length $\mu = 156$ bp) more slowly than the simBA-5 ($\mu = 100$ bp) or HiSeq ($\mu = 92$ bp) reads. We also performed a speed comparison against LMAT using one of the real samples discussed in LMAT's published results; on this sample Kraken was 38.82 times faster than LMAT, and 7.55 times faster than a version of LMAT using a smaller database (Appendix C).

4.2.5 Other variants of Kraken

To obtain maximal speed, Kraken needs to avoid page faults (instances where data must be brought from a hard drive into physical memory), so it is important that Kraken be run on a computer with enough RAM to hold the entire database. Although Kraken's default database requires 70 GB of RAM, we also developed a method to remove k-mers from the database that dramatically reduces the memory requirements. When using Kraken with this smaller database, we call it "MiniKraken"; for our results here, we use a 4 GB database. Compared to Kraken, the ability of MiniKraken to recognize species from short reads is lower, with sensitivity on our real sequence metagenomes dropping approximately 11% (Figure 13 and Table 13). On the high-error simBA-5 metagenome, MiniKraken's sensitivity was more than 25 percentage points lower than Kraken's, indicating that for short reads, high error rates can cause substantial loss in sensitivity. However, for all three metagenomes, MiniKraken was more precise than Kraken.

MiniKraken's high precision demonstrates that in many cases we do not need to examine all k-mers in a sequence to get the correct classification. Taking this idea to its extreme, we developed a "quick operation" mode for Kraken (and MiniKraken), where instead of querying all k-mers in a sequence against our database, we instead stop at the first k-mer that exists in the database, and use the LCA associated with that k-mer to classify the sequence. This operation mode (denoted by appending "-Q" to the classifier name) allows Kraken to skip tens or hundreds of k-mer queries per sequence, significantly increasing its classification speed with only a small drop in accuracy (Figure 13 and Table 13). Because a

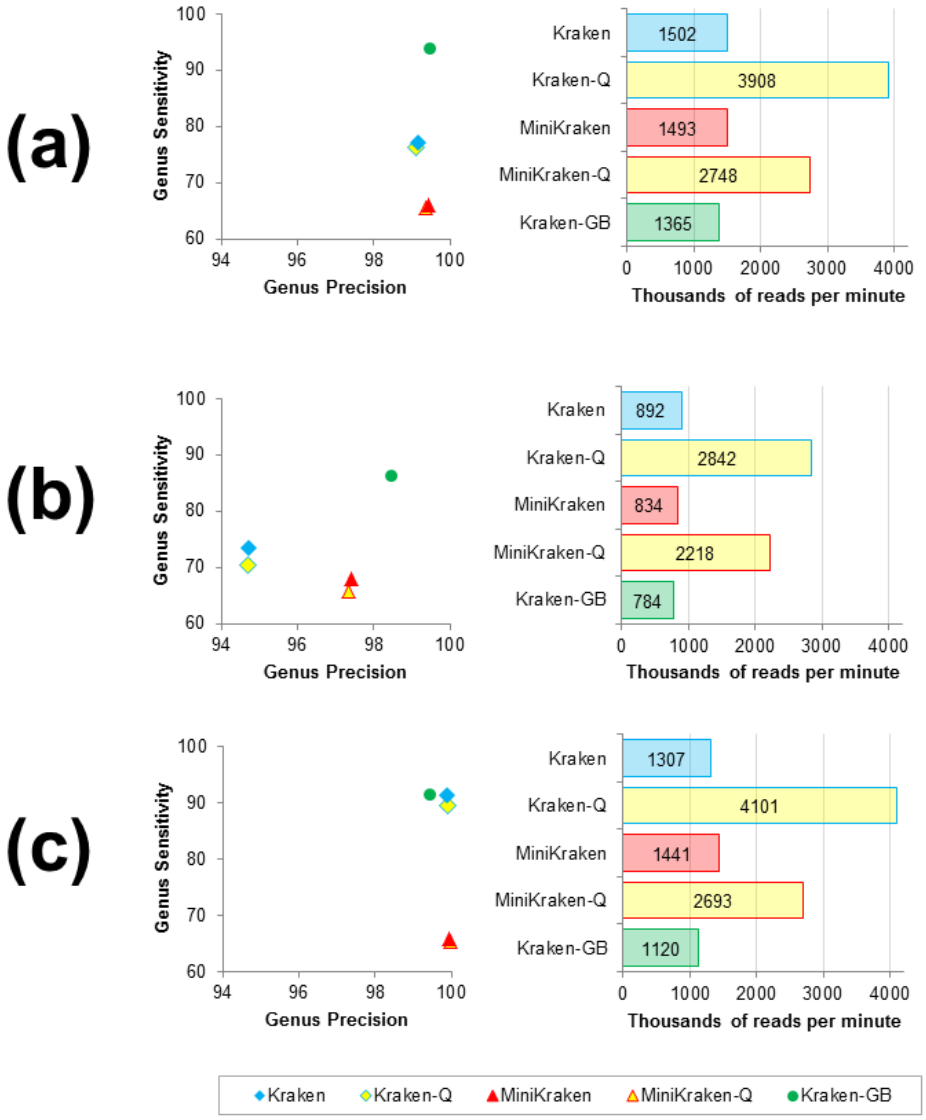

Figure 13. Classification accuracy and speed comparison of variants of Kraken across three simulated metagenomes. For each metagenome, genus precision and sensitivity are shown for five classifiers, and speed is shown for Kraken, along with a reduced memory version of Kraken (MiniKraken), quick execution versions of both (Kraken-Q and MiniKraken-Q), and Kraken run with a database containing draft and completed microbial genomes in GenBank (Kraken-GB). Results shown are from performance on the same metagenomes used in Figure 12. Note that scales of the axes differ from Figure 12, as the precision and speed of Kraken (and its variants) exceed that of the other classifiers used.

database containing fewer k-mers requires more queries from a sequence to find a hit, MiniKraken-Q is slower than Kraken-Q, even when MiniKraken is faster than Kraken.

We also created a variant Kraken database that contains GenBank's draft and completed genomes for bacteria and archaea, which we call Kraken-GB. (The regular version of Kraken only includes RefSeq complete genomes, numbering 2,256, while Kraken-GB contains 8,517 genomes.) Our hypothesis was that Kraken-GB would have a higher sensitivity than standard Kraken on our metagenomes, by virtue of its larger database. Comparing Kraken-GB's sensitivity for the HiSeq and MiSeq metagenomes to Kraken's, we see a large increase for Kraken-GB (Figure 13 and Table 13), primarily due to the presence of two genomes in these simulated metagenomic samples that have close relatives only in Kraken-GB's database (Methods).

Although Kraken-GB does have higher sensitivity than Kraken, it sometimes makes surprising errors, which we discovered were caused by contaminant and adapter sequence in the contigs of some draft genomes. These contaminant sequences come from other bacteria, viruses, or even human genomes, and they result in incorrectly labelled k-mers in the database. We attempted to remove these from Kraken-GB (Methods), but some contaminants may still slip through any filters. Thus for now, the default version of Kraken uses only complete RefSeq genomes.

4.2.6 Clade exclusion experiments

An important goal of metagenomics is the discovery of new organisms, and proper classification of novel organisms is a challenge for any classifier. Although a classifier cannot possibly give a novel species the proper species label, it may be able to identify the correct genus. To simulate the presence of novel organisms,

Measured Rank	Excluded rank					
	Species	Genus	Family	Order	Class	Phylum
Kingdom	100/24.4/24.4	100/7.9/7.9	100/2.8/2.8	100/2.3/2.3	100/1.5/1.5	100/1.1/1.1
Phylum	99.9/23.9/24.5	99.6/7.2/7.9	98.7/2.5/2.8	98.0/1.6/2.4	96.8/1.2/1.7	--
Class	99.7/24.7/25.3	99.1/7.1/7.9	96.7/2.0/3.0	93.2/1.0/2.3	--	--
Order	99.7/24.1/25.3	98.9/6.8/8.5	96.4/2.0/3.4	--	--	--
Family	99.7/25.4/26.7	98.5/8.5/10.8	--	--	--	--
Genus	99.2/26.3/33.2	--	--	--	--	--

Table 14. Classification statistics with clade exclusion for Kraken on the simBA-5 metagenome. For each measured rank, Kraken's rank-level precision, sensitivity, and classification percentage are shown. Classification percentage is defined here as the percentage of reads, with taxonomic entries at both the measured and excluded ranks, that were classified by Kraken with the clade of origin excluded.

we re-analyzed the simBA-5 metagenome after first removing organisms from the Kraken database that belonged to the same clade. I.e., for each read, we masked out database hits for the species of the read's origin, and evaluated Kraken's accuracy at the higher ranks (e.g., genus, family). We continued this masking and evaluation process for clades of origin up to the phylum rank. This procedure approximates how Kraken would classify the metagenomic reads if that clade were not present in the database.

Table 14 contains the results of this analysis. Kraken exhibited high rank-level precision in all cases where a clade is excluded, with rank-level precision remaining at or above 93% for all pairs of measured and excluded ranks. However, sensitivity was dramatically lower: at best, Kraken is able to classify approximately 33% of reads when their species has never been seen before. This is not surprising in light of Kraken's reliance on exact matches of relatively long k-mers: sequences deriving from different genera rarely share long exact matches. Nonetheless, the high precision in this experiment indicates that when Kraken is presented with novel organisms, it is likely either to properly classify them at higher levels or not classify them at all.

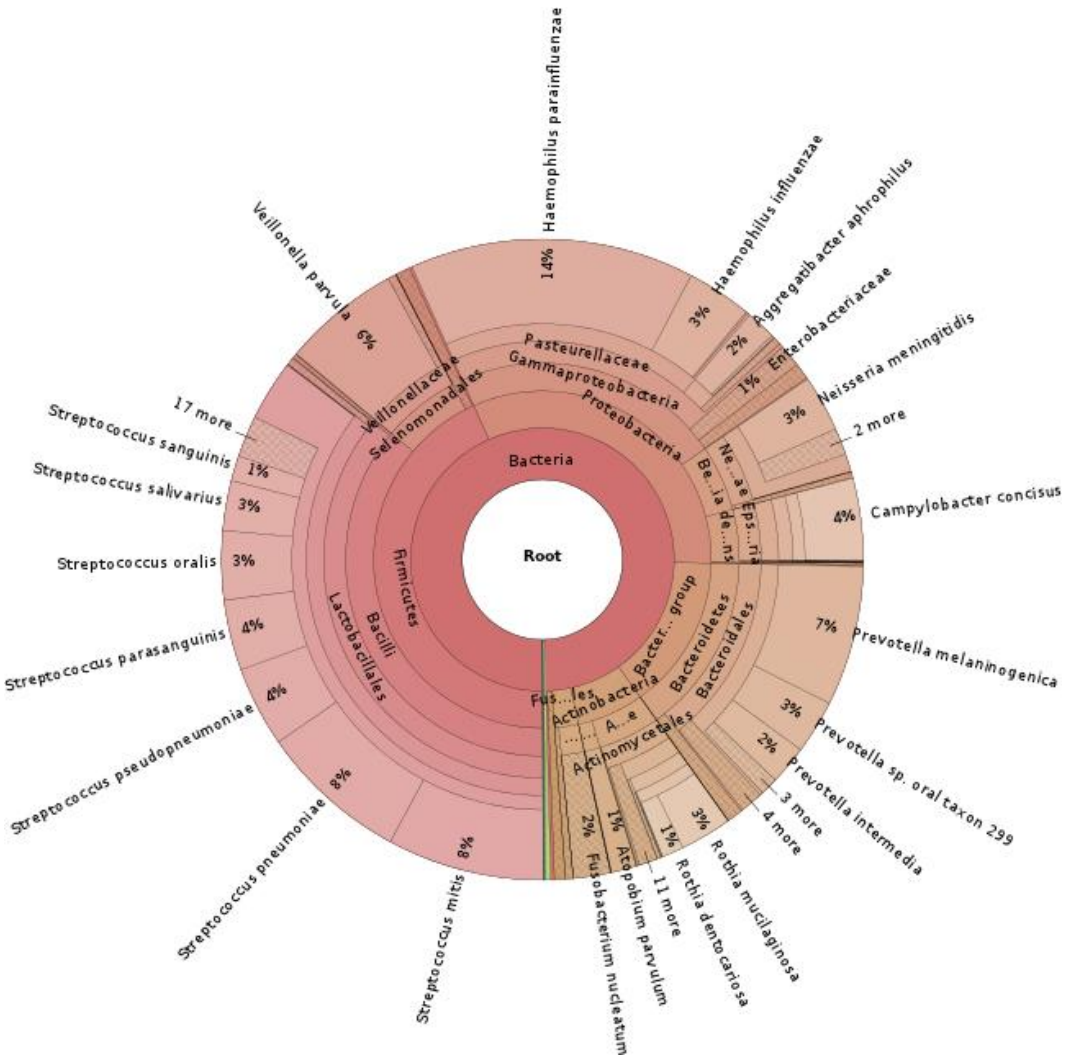

Figure 14. Taxonomic distribution of saliva microbiome reads classified by Kraken. Sequences from saliva samples collected from three individuals were classified by Kraken. The distribution of those reads that were classified by Kraken is shown.

4.2.7 Human Microbiome Project data

We used Kraken to classify reads from three saliva samples collected as part of the Human Microbiome Project. Because these samples were obtained from humans, we created a Kraken database containing bacterial, viral, and human genomes to classify these reads. Combining the three samples together, we report the taxonomic distribution of the classified reads (Figure 14). An analysis of the classified reads from the combined samples reveals that a majority of those reads

were classified into one of three genera: *Streptococcus* (30%), *Haemophilus* (17%), and *Prevotella* (13%). *Streptococcus mitis* [85], *Haemophilus parainfluenzae* [86], and *Prevotella melaninogenica* [87], the most abundant species (by read count) of each of these three genera, are all known to be associated with human saliva.

Of note is that 68.2% of the reads were not classified by Kraken. To determine why these reads were not classified by Kraken, we aligned a randomly-selected subset of 2,500 of these unclassified reads to the RefSeq bacterial genomes using BLASTN. Only 11% (275) of the subset of unclassified reads had a BLASTN alignment with an E-value $\leq 10^{-5}$ and identity $\geq 90\%$. This suggests that the vast majority of the reads unclassified by Kraken were significantly different from any known species, and thus simply impossible to identify.

4.3 Conclusions

Kraken's accuracy is comparable to that of Megablast for classifying short sequence reads, as might be expected given that both require long exact sequence matches (Kraken requires 31 bp exact matches, while Megablast requires 28 bp [9]). As we showed on the simBA-5 metagenome, where high sequence error rates were simulated, Megablast's inexact alignment strategy allowed it to tolerate more errors and achieve higher sensitivity than Kraken, which uses only exact alignment. We note that even in the face of this high error, Kraken's sensitivity still exceeded 90%, and its precision was 99.9%. With Kraken's high precision, users concerned with maximizing sensitivity could run Kraken first, and then run

another classification program on the reads not classified by Kraken, obtaining high sensitivity results much faster than with a single program.

An important constraint with Kraken is its memory usage: at present, the default database requires 70 GB, a value that will grow in linear proportion to the number of distinct k-mers in the genomic library (the database's records occupy 12 bytes per k-mer). For comparison, the only other k-mer based classifier, LMAT, reports a far larger database size of 619 GB. Where Kraken only stores the LCA for each k-mer, LMAT also records all genomes associated with a k-mer, resulting in a record size bounded only by the number of genomes in the library. The use of the reduced database in MiniKraken offers a nearly equivalent alternative should Kraken's database size be too large for available computational resources.

One important potential alternative use of Kraken is to rapidly identify contaminant sequences. As we noted, some of the draft microbial genomes in GenBank contain contaminating sequences from many different sources. Using a fast classifier like Kraken can quickly identify many such contaminants before they are included a draft assembly. Similarly, for microbial samples collected from humans, a Kraken database can be created that will quickly identify contaminating human reads from a metagenomic sample.

Finally, Kraken's results demonstrate the high speed and accuracy that is achievable through the use of short exact alignments. The Kraken database structure, which is tuned to rapidly query overlapping k-mers, enables Kraken to produce its results even faster than it would without the database facilitating this

type of query activity. We believe that this structure can find a use in other applications beyond taxonomic classification; for example, de Bruijn graphs, commonly used in genome assembly programs, can effectively be traversed by querying a database with overlapping k-mers [88], and that process can be made faster through the caching behavior of the Kraken database. Likewise, most operations that require querying overlapping k-mers should be able to run significantly faster through use of a data structure like the Kraken database.

4.4 Methods

4.4.1 Sequence classification algorithm

To classify a DNA sequence *S*, we collect all k-mers within that sequence into a set, denoted as $K(S)$. We then map each k-mer in $K(S)$, using the algorithm described below, to the lowest common ancestor (LCA) taxon of all genomes that contain that k-mer. These LCA taxa and their ancestors in the taxonomy tree form what we term the *classification tree*, a pruned subtree that is used to classify *S*. Each node in the classification tree is weighted with the number of k-mers in $K(S)$ that mapped to the taxon associated with that node. Then, each root-to-leaf (RTL) path in the classification tree is scored by calculating the sum of all node weights along the path. The maximum scoring RTL path in the classification tree is the *classification path*, and *S* is assigned the label corresponding to its leaf (in the case of multiple maximally scoring paths, the LCA of all those paths' leaves is selected). This algorithm, illustrated in Figure 11, allows Kraken to consider each k-mer within a sequence as a separate piece of evidence, and then attempt to resolve any conflicting evidence if necessary. Note that for an appropriate choice

of k, most k-mers will map uniquely to a single species, greatly simplifying the classification process. Sequences for which none of the k-mers in $K(S)$ are found in any genome are left unclassified by this algorithm.

The use of RTL path scoring in the classification tree is necessary in light of the inevitable differences between the sequences to be classified and the sequences present in any library of genomes. Such differences can, even for large values of *k*, result in a k-mer that is present in the library but associated with a species far removed from the true source species. By scoring the various RTL paths in the classification tree, we can compensate for these differences and correctly classify sequences even when a small minority of k-mers in a sequence would indicate that the sequence should be assigned an incorrect taxonomic label.

4.4.2 Database creation

Efficient implementation of Kraken's classification algorithm requires that the mapping of k-mers to taxa be performed by querying a pre-computed database. Kraken creates this database through a multi-step process, beginning with the selection of a library of genomic sequences. Kraken includes a default library, based on completed microbial genomes in the NCBI RefSeq database, but the library can be customized as needed by individual users.

Once the library is chosen, we use the Jellyfish multithreaded k-mer counter [89] to create a database containing every distinct 31-mer in the library. Once the database is complete, the 4-byte spaces Jellyfish used to store the k-mer counts in the database file are instead used by Kraken to store the taxonomic ID numbers of

the k-mers' LCA values. After the database is created by Jellyfish, the genomic sequences in the library are processed one at a time: for each sequence, the taxon associated with it is used to set the stored LCA values of all k-mers in the sequence. As sequences are processed, if a k-mer from a sequence has had its LCA value previously set, then the LCA of the stored value and the current sequence's taxon is calculated and that LCA is stored for the k-mer. Taxon information is obtained from the NCBI taxonomy database.

4.4.3 Database structure and search algorithm

Because Kraken very frequently uses a k-mer as a database query immediately after querying an adjacent k-mer, and because adjacent k-mers share a substantial amount of sequence, we utilize the minimizer concept [35] to group similar k-mers together. To explain our application of this concept, we here define the canonical representation of a DNA sequence *S* as the lexicographically smaller of *S* and the reverse complement of *S*. To determine a k-mer's minimizer of length *M*, we consider the canonical representation of all *M*-mers in the k-mer, and select the lexicographically smallest of those *M*-mers as the k-mer's minimizer. In practice, adjacent k-mers will often have the same minimizer.

In Kraken's database, all k-mers with the same minimizer are stored consecutively, and are sorted in lexicographical order of their canonical representations. A query for a k-mer *R* can then be processed by looking up in an index the positions in the database where the k-mers with *R*'s minimizer would be stored, and then performing a binary search within that region (Figure 15). Because adjacent k-mers often have the same minimizer, the search range is often

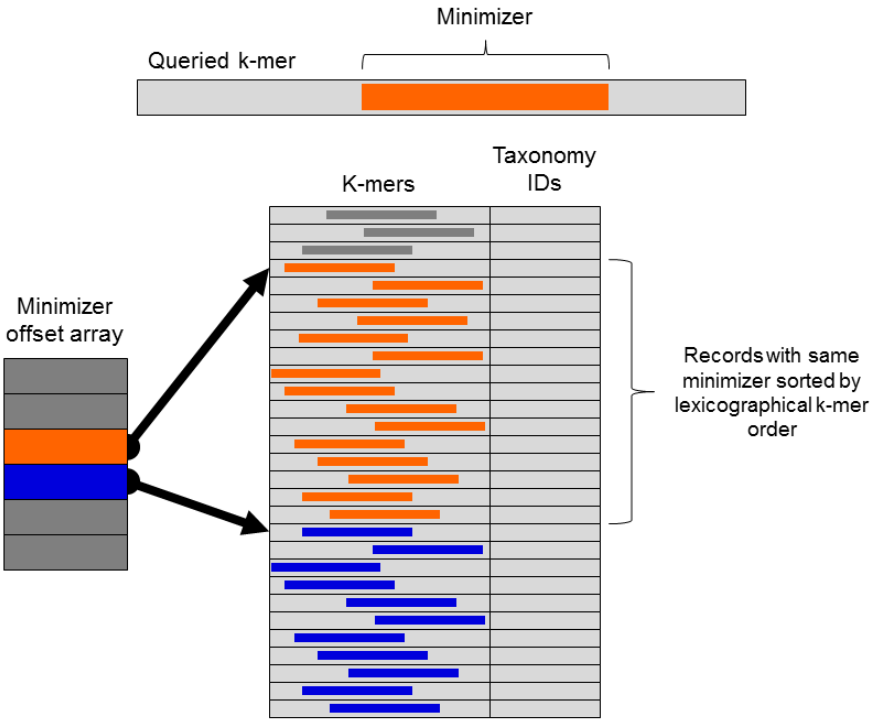

Figure 15. Kraken database structure. Each k-mer to be queried against the database has a specific substring that is its minimizer. To search for a k-mer in the database, the positions in the database that contain k-mers with the same minimizer are examined; these positions are quickly found by examining the minimizer offset array for the start positions of records with the k-mer's minimizer (orange) and the next possible minimizer (blue). Within a range of records associated with a given minimizer, records are sorted by lexicographical ordering of their k-mers, allowing a query to be completed using binary search over this range.

the same between two consecutive queries, and the search in the first query can often bring data into the CPU cache that will be used in the second query. By allowing memory accesses in subsequent queries to access data in the CPU cache instead of RAM, this strategy makes subsequent queries much faster than they would otherwise be. The index containing the offsets of each group of k-mers in the database requires 8 x 4^M bytes. By default Kraken uses 15 bp minimizers, but the user can modify this value; for example, in creating MiniKraken, we used 13 bp minimizers to ensure the total database size stayed under 4 GB.

In implementing Kraken, we made further optimizations to the structure and search algorithm described above. First, as noted by Roberts *et al*. [35], a simple lexicographical ordering of *M*-mers can result in a skewed distribution of minimizers that over-represents low-complexity *M*-mers. In Kraken, such a bias would create many large search ranges that would require more time to search. To create a more even distribution of minimizers (and thus speed up searches), we use the exclusive-or (XOR) operation to toggle half of the bits of each *M*-mer's canonical representation prior to comparing the *M*-mers to each other using lexicographical ordering. This XOR operation effectively scrambles the standard ordering, and prevents the large bias toward low-complexity minimizers.

We also take advantage of the fact that the search range is often the same between queries to make Kraken's queries faster. Rather than compute the minimizer each time we perform a query, we first search the previous range. If our queried k-mer is found in this range, the query can return immediately. If the k-mer is not found, then the minimizer is computed; if the k-mer's minimizer is the same as the last queried k-mer's, then the query fails, as the minimizer's search space has been shown not to have the k-mer. Only if the minimizer has changed does Kraken have to adjust the search range and search again for the k-mer.

4.4.4 Constructing simulated metagenomes

The HiSeq and MiSeq metagenomes were built using 20 sets of bacterial whole-genome shotgun reads. These reads were found either as part of the GAGE-B project [90] or in the NCBI Sequence Read Archive. Each metagenome contains sequences from 10 genomes (Table 15). Both the 10,000 and 10 million read

Metagenome	Genome	Source
HiSeq	*Aeromonas hydrophila* SSU	GAGE-B web site
HiSeq	*Bacillus cereus* VD118	GAGE-B web site
HiSeq	*Bacteroides fragilis* HMW615	GAGE-B web site
HiSeq	*Mycobacterium abscessus* 6G-0125-R	GAGE-B web site
HiSeq	*Pelosinus fermentans* A11	SRA run SRR515982
HiSeq	*Rhodobacter sphaeroides* 2.4.1	GAGE-B web site
HiSeq	*Staphylococcus aureus* M0927	GAGE-B web site
HiSeq	*Streptococcus pneumoniae* TIGR4	SRA run SRR387337
HiSeq	*Vibrio cholerae* CP1032(5)	GAGE-B web site
HiSeq	*Xanthomonas axonopodis pv. Manihotis* UA323	GAGE-B web site
MiSeq	*Bacillus cereus* VD118	GAGE-B web site
MiSeq	*Citrobacter freundii* 47N	SRA run SRR493656
MiSeq	*Enterobacter cloacae*	SRA run SRR568037
MiSeq	*Klebsiella pneumoniae* NES14	SRA run SRR493683
MiSeq	*Mycobacterium abscessus* 6G-0125-R	GAGE-B web site
MiSeq	*Proteus vulgaris* 66N	SRA run SRR493654
MiSeq	*Rhodobacter sphaeroides* 2.4.1	GAGE-B web site
MiSeq	*Staphylococcus aureus* ST22	SRA run ERR103400
MiSeq	*Salmonella enterica* Montevideo str. N19965	SRA run SRR493338
MiSeq	*Vibrio cholerae* CP1032(5)	GAGE-B web site

Table 15. Component genomes in the HiSeq and MiSeq simulated metagenomes. Some data were obtained from the GAGE-B project web site (http://ccb.jhu.edu/gage_b/), while others were found through searches of the NCBI Sequence Read Archive (SRA).

samples of each of these metagenomes had 10% of their sequences selected from each of the 10 component genome datasets (i.e., each genome had equal sequence abundance). All sequences were trimmed to remove low quality bases and adapter sequence.

The composition of these two metagenomes poses certain challenges to our classifiers. For example, *Pelosinus fermentans*, found in our HiSeq metagenome, cannot be correctly identified at the genus level by Kraken (or any of the other previously described classifiers), because there are no *Pelosinus* genomes in the RefSeq complete genomes database; however, there are 7 such genomes in Kraken-GB's database, including 6 strains of *P. fermentans*. Similarly, in our MiSeq metagenome, *Proteus vulgaris* is often classified incorrectly at the genus level because the only *Proteus* genome in Kraken's database is a single *Proteus*

mirabilis genome; 5 more *Proteus* genomes are present in Kraken-GB's database, allowing Kraken-GB to better classify reads from that genus. In addition, the MiSeq metagenome contains 5 genomes from the Enterobacteriaceae family (*Citrobacter*, *Enterobacter*, *Klebsiella*, *Proteus*, and *Salmonella*). The high sequence similarity between the genera in this family can make distinguishing between genera difficult for any classifier.

The simBA-5 metagenome was created by simulating reads from the set of complete bacterial and archaeal genomes in RefSeq. Replicons from those genomes were used if they were associated with a taxon that had an entry associated with the genus rank, resulting in a set of replicons from 607 genera. We then used the Mason read simulator (http://www.seqan.de/projects/mason/) with its Illumina model to produce 10 million 100-bp reads from these genomes. First we created simulated genomes for each species, using a SNP rate of 0.1% and an indel rate of 0.1% (both default parameters), from which we generated the reads. For the simulated reads, we multiplied the default mismatch and indel rates by 5, resulting in an average mismatch rate of 2% (ranging from 1% at the beginning of reads to 6% at the ends) and an indel rate of 1% (0.5% insertion probability and 0.5% deletion probability). For the simBA-5 metagenome, the 10,000 read set was generated from a random sample of the 10 million read set.

4.4.5 Evaluation of accuracy and speed

We elected to measure accuracy primarily at the genus level, which was the lowest level for which we could easily determine the taxonomy information for PhymmBL and NBC's predictions in an automated fashion. (This is due to the

manner in which PhymmBL and NBC report their results.) Because some genomes do not have taxonomic entries at all seven ranks (species, genus, family, order, class, phylum, and kingdom), we defined genus-level sensitivity as A/B, where A was the number of reads with an assigned genus that were correctly classified at that rank, and B was the total number of reads with any assigned genus. We defined sensitivity similarly for other taxonomic ranks.

Because Kraken may classify a read at levels above the species, measuring its precision requires us to define the effect on precision of assigning the correct genus (for example) while not assigning a species at all. For this reason, we defined rank-level precision as C/(D+E), where C was the number of reads labeled at or below correct taxon at the measured rank, D was the number of reads labeled at or below the measured rank, and E was the number of reads incorrectly labeled above the measured rank. For example, given a read *R* that should be labeled as *Escherichia coli*: a labeling of *R* as *E. coli*, *E. fergusonii*, or *Escherichia* would improve genus-level precision; a label of Enterobacteriaceae (correct family) or Proteobacteria (correct phylum) would have no effect on genus-level precision; and a label for *R* of *Bacillus* (incorrect genus) or Firmicutes (incorrect phylum) would decrease genus-level precision.

When evaluating PhymmBL's accuracy, following its developers' advice [81], we selected a genus confidence threshold for our comparisons. We selected 3,333 reads from the simulated medium complexity (simMC) [91] dataset, covering 31 different genera. To simulate short reads from the Sanger sequence data in the simMC set, we selected the last 100 bp from each of the reads. We then ran

Confidence Level	Accuracy	Precision	Sensitivity
0.0	54.9	54.9	54.9
(results equal for all confidence levels from 0-0.45)			
0.45	54.9	54.9	54.9
0.50	55.2	55.5	54.9
0.55	66.8	86.1	54.5
0.60	69.6	97.5	54.2
0.65	69.7	98.0	54.1
0.70	69.5	98.2	53.8
0.75	68.6	98.2	52.7
0.80	61.8	98.1	45.1
0.85	26.4	97.7	15.3
0.90	0.1	100.0	0.0
0.95	No labels at or above this confidence level		

Table 16. PhymmBL classification accuracy across different confidence levels. Genus-level accuracy for PhymmBL on 3,333 reads from the simMC dataset is shown, with varying genus confidence thresholds applied. Note that when only labels with genus confidence >= 0.9 were considered, only 1 label remained.

PhymmBL against those 100 bp reads, and then evaluated the genus-level sensitivity and precision of PhymmBL's predictions with genus confidence thresholds from 0 to 1, in increments of 0.05. We found that a threshold of 0.65 yielded the highest F-score value (the harmonic mean of sensitivity and precision), with 0.60 and 0.70 also having F-scores within 0.5 percentage points of the maximum (Table 16). We therefore used the 0.65 genus confidence threshold in our comparisons; although the selection of a threshold depends on a user's individual needs, and so is to some extent arbitrary, a threshold selected in this manner provides a more proper comparison to a selective classifier such as Kraken than no threshold at all.

The time and accuracy results when using Megablast as a classifier were obtained from the log data produced by PhymmBL, as PhymmBL uses Megablast for its alignment step. When assigning a taxonomic label to a read with Megablast, we

used the taxon associated with the first reported alignment. Megablast was run with default options.

Speed was evaluated by using single-threaded operation of each program (except for NBC); PhymmBL was altered to have its call to the "blastn" program use one thread instead of two. NBC was run with 36 concurrent processes operating on disjoint sets of genomes in its genomic library, and total time for the classifier was determined by summing the decompression and scoring times for each genome. Wall clock times were recorded for all classifiers. In comparing Kraken to the other classifiers, we used BLAST+ 2.2.27, PhymmBL 4.0, NBC 1.1, and MetaPhlAn 1.7.6. Classifiers were all run on the same computer, with 48 AMD Opteron 6172 2.1 GHz CPUs and 252 GB of RAM, running Red Hat Enterprise Linux 5. The datasets used for speed evaluation had 10,000 reads each for all programs other than Kraken (and its variants) and MetaPhlAn, which used 10,000,000 read datasets; the increased read numbers were used with these faster programs to minimize the impact of initial and final operations that take place during the programs' execution.

Although Kraken is the only one of the programs we examined that explicitly performs operations to ensure its data is in physical memory before classification, we wanted to be sure that all programs were evaluated in a similar manner. For each program, we read all database files (e.g. IMM files and BLAST databases for PhymmBL, k-mer frequency lists for NBC, the Bowtie index for MetaPhlAn) into memory three times before running the program for the purposes of the speed

evaluation, in order to place the database content in the operating system cache (which is stored in physical memory).

4.4.6 Reduced database sizes

To generate the 4 GB database for our MiniKraken results, we removed the first 18 of every block of 19 records in the standard Kraken database. A shrinking factor of 19 was selected as it was the smallest integer factor that would reduce the size to less than 4 GB, a size that could easily fit into memory on many common personal computers. For users that have more RAM available, Kraken allows a smaller shrinking factor to be used, which will allow increased sensitivity.

4.4.7 Use of draft genomes

When constructing the Kraken-GB database, we noticed several contigs that had known adapter sequences at the ends of the contigs. In subsequent tests, we also found that some sequences in samples with large amounts of human sequence were consistently misclassified by this database, leading us to conclude that contamination was likely present in the draft genomes. In an attempt to counteract the presence of this contamination, we removed from the database those k-mers from known adapter sequences, as well as the first and last 20 k-mers from each of the draft contigs. While this did improve classification, it did not eliminate the misclassification problem. For this reason, we believe that if draft genomes are used in a Kraken database, very stringent measures should be used to remove contaminant sequence from the genomic library.

4.4.8 Clade exclusion experiments

When re-analyzing the simBA-5 dataset for our clade exclusion experiments, some reads were not used for certain pairs of measured and excluded ranks. If a read's origin lacked a taxonomic entry at either of the measured or excluded ranks, it was not used for that particular experiment.

In addition, a read was not used in an experiment unless at least two other taxa represented in our database (aside from the excluded clade) at the excluded rank shared the clade of origin's taxon at the measured rank. For example, a read from genus G would not be used in an experiment measuring accuracy at the class rank and excluding the genus rank unless G's home class had at least two other genera with genomes in Kraken's genomic library. Without this filtering step, were a genus excluded when it was the only genus in its class, Kraken could not possibly name the correct class, as all entries in the database from that class could be excluded as well. This is the same approach taken in similar experiments that were used to evaluate PhymmBL [22].

4.4.9 Human microbiome data classification

We classified the HMP data using a Kraken database made from complete RefSeq bacterial, archaeal, and viral genomes, along with the GRCh37 human genome. We retrieved the sequences of three accessions (SRS019120, SRS014468, and SRS015055) from the NCBI Sequence Read Archive, and each accession had two runs submitted. All reads were trimmed to remove low quality bases and adapter sequence. Krona [92] was used to generate all taxonomic distribution plots.

Because the sequences were all paired reads, we joined the reads together by concatenating the mates with a sequence of ‘NNNNN’ between them. Kraken ignores k-mers with ambiguous nucleotides, so the k-mers that span these ‘N’ characters do not affect classification. This operation allowed Kraken to classify a pair of reads as a single unit rather than having to classify the mates separately.

5 Conclusions

The work in this dissertation contributes advances toward improving the accuracy of microbial genome annotation and the classification speed of metagenomic sequences. Both of these problems have gained importance in recent years due to the increases in sequencing throughput and completed microbial genomes.

Chapter 2 introduced a pipeline that found over 10,000 genes that were wrongly omitted from the annotation of microbial genomes in GenBank. The pipeline uses open source and widely available tools, allowing researchers to easily add the pipeline's steps to their own annotation pipeline. The pipeline could also easily be integrated into the GenBank submission process to help keep incorrect information from ever entering the public databases.

In Chapter 3, I discussed Phantim, a tool that uses multiple protein sequence alignments to improve gene start site annotation. Phantim's high precision helped to reveal a link between an annotation's high use of rare start codons and incorrect start site annotation. The degree of rare start codon usage is easily calculated, and can also be added to the GenBank submission process to improve the quality of incoming annotations.

Chapter 4 focuses on the problem of taxonomic classification of metagenomic sequences. In that chapter I introduced Kraken, a taxonomic classifier that is both highly accurate and very fast. This combination of attributes filled an important need in metagenomic research, as no previous program had been able to accomplish both high accuracy and high speed. In Kraken's case, its accuracy is

comparable to the best existing methods, and its speed is faster than all previous methods, including those that sacrifice sensitivity for speed.

Kraken has also found use in other areas of bioinformatics aside from metagenomics. In DIAMUND [93], a variant detection pipeline for exome data, there is a step in the pipeline where thousands of "interesting" k-mers and millions of reads are processed to select a set of reads that have at least one of the k-mers of interest. Kraken's modules were easily repurposed to perform this task, turning the taxonomic classifier into a binary classifier. The addition of Kraken to the DIAMUND pipeline reduced the time needed for this binary classification task significantly; for one data set, the runtime was reduced from 2 hours to 10 minutes.

As sequencing technology continues to improve, the ability of computational methods to handle the challenges posed by this improvement gains in importance. The results obtained by the pipelines in chapters 2 and 3 demonstrate a clear need for validation of genome annotations prior to their insertion into our genomic databases. The pipelines described aid in performing that validation. The speed provided by Kraken, achieved while maintaining high sensitivity, will enable new avenues of metagenomic research. The work presented in this dissertation provides researchers with tools to address current challenges of bioinformatics, and it is my hope that this work will provide the foundation for future developments in the field as well.

A Results of examination of Phantim-only genes

This appendix contains a list of all Phantim predictions that do not have a 3' match in a genome's RefSeq annotation, as well as a recommendation regarding any changes in the annotation with respect to the predicted gene.

Table 17. List of Phantim 3' disagreements with RefSeq annotations

Genome	Start	Stop	Str.	Recommendation
A. citrulli	2978649	2978326	rev.	**Add** gene to annotation. This gene is an exact nucleotide copy of Aave_2936. This gene only overlaps one gene, Aave_2709, by 4 nt.
A. radiobacter chr. 1	579295	579161	rev.	**Add** gene to annotation. This gene has over 100 BLASTP hits in nr, all to ribosomal proteins; one hit is a 94% amino acid identity alignment, with 100% coverage, to a "50S ribosomal protein" in *Methylobacterium populi* BJ001. Its region is intergenic according to the RefSeq annotation.
A. radiobacter chr. 1	1323593	1323829	fwd.	**Replace** Arad_1670. This gene is almost completely overlapped by Arad_1670, named an "acyl carrier protein". Arad_1670 has only 26 BLASTP hits in nr, and only 9 with E-values less than 1e-3. In comparison, this region has over 100 BLASTP hits, with the 100th best having an E-value of 5e-21; the name of all these hits is also "acyl carrier protein".
A. radiobacter chr. 1	1571695	1572003	fwd.	**Add** gene to annotation. This gene has over 100 BLASTP hits to ribosomal proteins in nr, with 100% amino acid identity to a "30S ribosomal protein S10" in *Rhizobium etli*. Its region is intergenic according to the RefSeq annotation.

A. radiobacter chr. 1	1669050	1668718	rev.	**Replace** Arad_ 2117. 77 nt overlap with Arad_2117, a hypothetical protein with only 3 BLASTP hits in nr, the best having an E-value of 2.1. This gene has over 100 BLASTP hits in nr, one with 88% amino acid identity to an "iron-sulfur cluster assembly accessory protein" in *Rhizobium leguminosarum*.
A. radiobacter chr. 2	1069411	1069803	fwd.	Add gene to annotation. This gene has 101 BLASTP hits to genes in nr, including an 87% amino acid identity alignment to a "glutathione-dependent formaldehyde-activating, GFA" gene in *Mesothizobium* sp. BNCI. Its region is intergenic according to the RefSeq annotation.
B. anthracis	3825729	3825601	rev.	Replace BA_4175. This gene is overlapped completely by BA_4175, which is a hypothetical protein with only 5 BLASTP hits in nr. This gene has 22 BLASTP hits to genes in various *Bacillus* and other genomes, including a 72% amino acid identity alignment to a phosphoesterase gene in *B. thuringiensis*. In addition, this gene has 81% amino acid identity to BA_4174, located just upstream.
Bradyrhizobium BTAi1	1946535	1946260	rev.	**Add** gene to annotation. This gene has 97 BLASTP hits in nr, most of which are alignments to conserved hypothetical proteins and proteins of unknown function (DUF1153). Its region is currently intergenic according to the RefSeq annotation.

M. tuberculosis	2765730	2765107	rev.	**Replace** MT2541. This gene has over 100 BLASTP hits in nr, including one with 57% amino acid identity to "DSBA oxidoreductase" in *Streptomyces* sp. SPB78. MT2541 is a hypothetical protein that completely overlaps this gene, and has only 17 BLASTP hits, only one of which has an E-value below 1.
M. tuberculosis	2943593	2943240	rev.	**Replace** MT2694. This gene has over 100 BLASTP hits in nr, including one with 98% coverage and 58% amino acid identity to a "Cupin 2 conserved barrel domain-containing protein" in *Nakamurella multipartita*. MT2694 is a hypothetical protein that completely overlaps this gene, and has only 2 BLASTP hits, with E-values of 2.3 and 7.3.
M. tuberculosis	4103085	4103603	fwd.	**Replace** MT3770. This gene has over 100 BLASTP hits in nr, including one with 100% amino acid identity to a "transmembrane protein" in *M. tuberculosis* H37Rv. MT3770 is a hypothetical protein that overlaps all but 72 nt of this gene, and has only 6 BLASTP hits, only one of which has an E-value below 1.
V. cholerae chr. I	1651882	1652085	fwd.	**Add** gene to annotation. This gene has over 100 BLASTP hits to genes in nr, two of which are "Zn-ribbon proteins" in *Idiomarina* genomes. Its region is intergenic according to the RefSeq annotation.

B Results of examination of Phantim/RefSeq 5' disagreements

This appendix contains a list of all Phantim predictions that have a 3' match but not a 5' match in a genome's RefSeq annotation, as well as notes regarding the conservation found in the predicted gene's multiple alignment. Evaluations indicate one or more of the following is true for the gene:

- **Change**: strong, clear conservation exists and supports the prediction; annotation should be changed.
- **ReviewWeak**: gene needs further examination, due to weaker evidence for the change; for genes annotated with a rare start codon, this can occur if the annotated start site is close (within ~15 nt) to the predicted start site.
- **ReviewUp**: gene needs further examination, as conservation exists upstream of predicted start (not just annotated start); this may be evidence of a frameshift or operon.
- **CTG** or **ATT**: a rare start codon (CTG/ATT) is annotated for this gene.
- **Keep**: strong conservation is present for the annotated start, and annotation should be left unchanged.

Table 18. Results of examination of Phantim and RefSeq 5' disagreements

		RefSeq			Phantim		
Genome	Gene ID	Start	Stop	Str.	Start	Extra (nt)	Evaluation
A. citrulli	Aave_0289	320917	321636	fwd.	320743	+174	**Change**
A. citrulli	Aave_0691	750181	752169	fwd.	749971	+210	**ReviewUp**
A. citrulli	Aave_2611	2860437	2863607	fwd.	2860347	+90	**Change**
A. citrulli	Aave_2652	2914135	2914890	fwd.	2914012	+123	**Change**
A. citrulli	Aave_3114	3439123	3439587	fwd.	3439066	+57	**Change**
A. citrulli	Aave_3366	3722691	3721507	rev.	3722721	+30	**ReviewUp**
A. citrulli	Aave_4383	4876301	4875558	rev.	4876328	+27	**Change**
A. citrulli	Aave_4645	5170949	5170674	rev.	5171006	+57	**Change**
A. radiob. chr. 1	Arad_0252	225153	224752	rev.	225234	+81	**Change**
A. radiob. chr. 1	Arad_0288	257347	256451	rev.	257692	+345	**ReviewWeak**

A. radiob. chr. 1	Arad_0306	270939	272090	fwd.	270489	+450	**ReviewUp**
A. radiob. chr. 1	Arad_0597	493707	494546	fwd.	493656	+51	**Change**
A. radiob. chr. 1	Arad_0772	635281	635916	fwd.	635233	+48	**Change CTG**
A. radiob. chr. 1	Arad_1495	1187396	1189285	fwd.	1187294	+102	**Change**
A. radiob. chr. 1	Arad_1676	1329161	1328331	rev.	1329158	-3	**ReviewWeak CTG**
A. radiob. chr. 1	Arad_1810	1431875	1431402	rev.	1431869	-6	**ReviewWeak CTG**
A. radiob. chr. 1	Arad_1862	1475491	1477029	fwd.	1475518	-27	**Change CTG**
A. radiob. chr. 1	Arad_1909	1517403	1518875	fwd.	1517394	+9	**Change CTG**
A. radiob. chr. 1	Arad_1938	1541374	1542411	fwd.	1541383	-9	**ReviewUp CTG**
A. radiob. chr. 1	Arad_2513	1986301	1987098	fwd.	1986265	+36	**Change**
A. radiob. chr. 1	Arad_2726	2165484	2165020	rev.	2165508	+24	**Change CTG**
A. radiob. chr. 1	Arad_2787	2210814	2209909	rev.	2210859	+45	**Change**
A. radiob. chr. 1	Arad_3282	2619023	2619982	fwd.	2619059	-36	**Change CTG**
A. radiob. chr. 1	Arad_3401	2727288	2726584	rev.	2727369	+81	**Change**
A. radiob. chr. 1	Arad_3755	2996226	2995261	rev.	2996292	+66	**Change**
A. radiob. chr. 1	Arad_4308	3444447	3442915	rev.	3444444	-3	**ReviewUp CTG**
A. radiob. chr. 1	Arad_4366	3491052	3490612	rev.	3491148	+96	**Change**
A. radiob. chr. 1	Arad_4394	3516274	3513554	rev.	3516244	-30	**Change CTG**
A. radiob. chr. 1	Arad_4606	3694076	3694975	fwd.	3694103	-27	**Change CTG**
A. radiob. chr. 1	Arad_4853	3926590	3926063	rev.	3926629	+39	**Change**
A. radiob. chr. 2	Arad_7499	430127	431005	fwd.	430130	-3	**ReviewWeak CTG**
A. radiob. chr. 2	Arad_7920	817597	818448	fwd.	817543	+54	**Change**
A. radiob. chr. 2	Arad_8127	988105	989187	fwd.	988120	-15	**Change CTG**
A. radiob. chr. 2	Arad_8281	1127905	1127375	rev.	1127959	+54	**Change**
A. radiob. chr. 2	Arad_8316	1158746	1159600	fwd.	1158749	-3	**ReviewWeak CTG**
A. radiob. chr. 2	Arad_8341	1183088	1183612	fwd.	1183064	+24	**Change**
A. radiob. chr. 2	Arad_8351	1190315	1191175	fwd.	1190318	-3	**ReviewWeak CTG**
A. radiob. chr. 2	Arad_8840	1594101	1595582	fwd.	1593927	+174	**Change**
A. radiob. chr. 2	Arad_9098	1816141	1817154	fwd.	1816096	+45	**Change**
A. radiob. chr. 2	Arad_9333	2023774	2024781	fwd.	2023777	-3	**ReviewWeak CTG**
A. radiob. chr. 2	Arad_9399	2087678	2087016	rev.	2087687	+9	**Change CTG**
A. radiob. chr. 2	Arad_9605	2271724	2270276	rev.	2271709	-15	**Change CTG**
B. anthracis	BA_0265	258100	257393	rev.	258142	+42	**Change**
B. anthracis	BA_5541	5032319	5031801	rev.	5032340	+21	**Change**
B. subtilis	BSU01460	151303	152133	fwd.	151264	+39	**Change**
B. subtilis	BSU25180	2599266	2598616	rev.	2599332	+66	**Change**
B. subtilis	BSU26550	2714140	2713949	rev.	2714209	+69	**Change**
B. subtilis	BSU28870	2953349	2952828	rev.	2953331	-18	**Keep ATT**
Bradyrhiz. BTAi1	BBta_0068	70735	69536	rev.	70759	+24	**Change**
Bradyrhiz. BTAi1	BBta_0178	184778	185194	fwd.	184751	+27	**Change**
Bradyrhiz. BTAi1	BBta_0228	234160	234495	fwd.	234097	+63	**Change**
Bradyrhiz. BTAi1	BBta_0456	456727	457239	fwd.	456577	+150	**Change**
Bradyrhiz. BTAi1	BBta_0472	470341	470496	fwd.	470215	+126	**Change**
Bradyrhiz. BTAi1	BBta_0740	761034	760111	rev.	761145	+111	**Change**
Bradyrhiz. BTAi1	BBta_0948	981763	982704	fwd.	981736	+27	**Change**
Bradyrhiz. BTAi1	BBta_0951	984964	984527	rev.	985012	+48	**Change**
Bradyrhiz. BTAi1	BBta_1240	1316169	1317290	fwd.	1316007	+162	**Change**
Bradyrhiz. BTAi1	BBta_1457	1543707	1543964	fwd.	1543659	+48	**Change**
Bradyrhiz. BTAi1	BBta_1599	1672857	1672570	rev.	1672893	+36	**Change**
Bradyrhiz. BTAi1	BBta_1707	1780631	1780266	rev.	1780661	+30	**Change**

Bradyrhiz. BTAi1	BBta_1788	1856837	1857526	fwd.	1856831	+6	**ReviewWeak CTG**
Bradyrhiz. BTAi1	BBta_2009	2074788	2075000	fwd.	2074764	+24	**Change**
Bradyrhiz. BTAi1	BBta_2165	2241836	2243518	fwd.	2241800	+36	**Change**
Bradyrhiz. BTAi1	BBta_2898	3004559	3005311	fwd.	3004529	+30	**Change**
Bradyrhiz. BTAi1	BBta_3040	3177551	3175974	rev.	3177626	+75	**ReviewWeak**
Bradyrhiz. BTAi1	BBta_3463	3614525	3613716	rev.	3614585	+60	**Change**
Bradyrhiz. BTAi1	BBta_3585	3745821	3747008	fwd.	3745767	+54	**Change**
Bradyrhiz. BTAi1	BBta_3602	3767174	3768643	fwd.	3767144	+30	**Change**
Bradyrhiz. BTAi1	BBta_3675	3838804	3840465	fwd.	3838771	+33	**Change**
Bradyrhiz. BTAi1	BBta_3755	3933201	3933923	fwd.	3933168	+33	**Change**
Bradyrhiz. BTAi1	BBta_3767	3947285	3946362	rev.	3947396	+111	**Change**
Bradyrhiz. BTAi1	BBta_3829	4015302	4012717	rev.	4015368	+66	**Change**
Bradyrhiz. BTAi1	BBta_4168	4379434	4378142	rev.	4379488	+54	**Change**
Bradyrhiz. BTAi1	BBta_4219	4432432	4433520	fwd.	4432330	+102	**ReviewUp**
Bradyrhiz. BTAi1	BBta_4257	4470093	4469278	rev.	4470129	+36	**Change**
Bradyrhiz. BTAi1	BBta_4324	4530792	4531415	fwd.	4530714	+78	**Change**
Bradyrhiz. BTAi1	BBta_4516	4731195	4730293	rev.	4731216	+21	**Change**
Bradyrhiz. BTAi1	BBta_5019	5249017	5249691	fwd.	5248993	+24	**Change**
Bradyrhiz. BTAi1	BBta_5059	5283790	5283476	rev.	5284099	+309	**Change**
Bradyrhiz. BTAi1	BBta_5089	5314187	5314029	rev.	5314220	+33	**Change**
Bradyrhiz. BTAi1	BBta_5101	5326283	5327206	fwd.	5326172	+111	**Change**
Bradyrhiz. BTAi1	BBta_5121	5349491	5348595	rev.	5349536	+45	**Change**
Bradyrhiz. BTAi1	BBta_5178	5407689	5408177	fwd.	5407632	+57	**Change**
Bradyrhiz. BTAi1	BBta_5409	5630107	5629340	rev.	5630137	+30	**ReviewWeak**
Bradyrhiz. BTAi1	BBta_6332	6594995	6595918	fwd.	6594884	+111	**Change**
Bradyrhiz. BTAi1	BBta_6527	6808355	6810403	fwd.	6808175	+180	**Change CTG**
Bradyrhiz. BTAi1	BBta_6829	7153662	7153033	rev.	7153683	+21	**Change**
Bradyrhiz. BTAi1	BBta_6844	7170752	7170844	fwd.	7170638	+114	**Change**
Bradyrhiz. BTAi1	BBta_7012	7346366	7346106	rev.	7346408	+42	**Change**
Bradyrhiz. BTAi1	BBta_7069	7410657	7411580	fwd.	7410546	+111	**Change**
Bradyrhiz. BTAi1	BBta_7206	7566745	7565036	rev.	7566808	+63	**Change**
Bradyrhiz. BTAi1	BBta_7442	7816925	7817194	fwd.	7816703	+222	**Change**
Bradyrhiz. BTAi1	BBta_7495	7870110	7869391	rev.	7870164	+54	**Change**
Bradyrhiz. BTAi1	BBta_7569	7953895	7953326	rev.	7953967	+72	**Change**
C. violaceum	CV_0086	97321	99189	fwd.	97279	+42	**Change**
C. violaceum	CV_0099	114821	113448	rev.	115055	+234	**ReviewUp**
C. violaceum	CV_0668	695894	696301	fwd.	695831	+63	**Change**
C. violaceum	CV_0744	764543	764809	fwd.	764453	+90	**Change**
C. violaceum	CV_0766	788326	786851	rev.	788383	+57	**Change**
C. violaceum	CV_0794	818319	817570	rev.	818394	+75	**Change**
C. violaceum	CV_1122	1175575	1176570	fwd.	1175503	+72	**Change**
C. violaceum	CV_1162	1221800	1220469	rev.	1221842	+42	**Change**
C. violaceum	CV_1224	1285455	1287536	fwd.	1285329	+126	**Change**
C. violaceum	CV_1457	1545716	1545402	rev.	1545752	+36	**Change**
C. violaceum	CV_3405	3699237	3699959	fwd.	3699075	+162	**Change**
C. violaceum	CV_3735	4033281	4033877	fwd.	4032915	+366	**ReviewUp**
C. violaceum	CV_3965	4289117	4286490	rev.	4289129	+12	**ReviewWeak CTG**
C. violaceum	CV_4100	4442560	4441790	rev.	4442584	+24	**ReviewWeak CTG**
C. violaceum	CV_4405	4750493	4750305	rev.	4750514	+21	**Change CTG**
E. coli	b0429	447270	446941	rev.	447351	+81	**Change**

E. coli	b0923	974845	975549	fwd.	974818	+27	**ReviewWeak**
E. coli	b1081	1136594	1137535	fwd.	1136564	+30	**Change**
E. coli	b1432	1501741	1502889	fwd.	1501681	+60	**Change**
E. coli	b1718	1798662	1798120	rev.	1798554	-108	**Keep ATT**
E. coli	b4461	2746796	2748082	fwd.	2746820	-24	**Keep CTG**
H. pylori	HP0885	935407	936792	fwd.	935332	+75	**Change**
H. pylori	HP1507	1580322	1581479	fwd.	1580280	+42	**Change**
H. salinarum	OE3470F	1285778	1286029	fwd.	1285745	+33	**Change**
M. tuberculosis	MT0055	52775	53188	fwd.	52754	+21	**Change**
M. tuberculosis	MT0436	510004	509207	rev.	510034	+30	**Change**
M. tuberculosis	MT0738	808614	810887	fwd.	808524	+90	**Change**
M. tuberculosis	MT0786	857867	856839	rev.	857966	+99	**Change**
M. tuberculosis	MT1041	1131690	1133303	fwd.	1131669	+21	**Change**
M. tuberculosis	MT2188	2390918	2389791	rev.	2391035	+117	**Change**
M. tuberculosis	MT2191	2393215	2392427	rev.	2393320	+105	**ReviewUp**
M. tuberculosis	MT3860	4192534	4192034	rev.	4192555	+21	**Change**
X. bovienii	XBJ1_0065	69975	72482	fwd.	69954	+21	**ReviewUp**
X. bovienii	XBJ1_0068	75246	77303	fwd.	75195	+51	**Change**
X. bovienii	XBJ1_0241	244359	243160	rev.	244416	+57	**Change**
X. bovienii	XBJ1_0338	346775	345549	rev.	346880	+105	**Change**
X. bovienii	XBJ1_0342	349583	353326	fwd.	349559	+24	**Change**
X. bovienii	XBJ1_0604	621716	625150	fwd.	621668	+48	**Change**
X. bovienii	XBJ1_0731	718904	721669	fwd.	718835	+69	**Change**
X. bovienii	XBJ1_0803	822940	822218	rev.	822967	+27	**ReviewWeak**
X. bovienii	XBJ1_0893	903055	904665	fwd.	902896	+159	**Change**
X. bovienii	XBJ1_1116	1139112	1140110	fwd.	1139070	+42	**Change**
X. bovienii	XBJ1_1800	1758537	1759787	fwd.	1758480	+57	**Change**
X. bovienii	XBJ1_1924	1859695	1861266	fwd.	1859635	+60	**Change**
X. bovienii	XBJ1_1926	1862240	1863208	fwd.	1862156	+84	**Change**
X. bovienii	XBJ1_1960	1892419	1893384	fwd.	1892389	+30	**ReviewWeak**
X. bovienii	XBJ1_2624	2590238	2590903	fwd.	2590196	+42	**Change**
X. bovienii	XBJ1_2675	2630754	2628988	rev.	2630820	+66	**Change**
X. bovienii	XBJ1_2786	2754576	2753962	rev.	2754792	+216	**ReviewUp**
X. bovienii	XBJ1_2841	2792338	2791175	rev.	2792377	+39	**Change**
X. bovienii	XBJ1_2847	2800875	2800045	rev.	2800932	+57	**ReviewWeak**
X. bovienii	XBJ1_2862	2811808	2810801	rev.	2811838	+30	**Change**
X. bovienii	XBJ1_2952	2918549	2919121	fwd.	2918501	+48	**Change**
X. bovienii	XBJ1_3332	3256054	3256938	fwd.	3256009	+45	**Change**
X. bovienii	XBJ1_3473	3382791	3382066	rev.	3382890	+99	**Change**
X. bovienii	XBJ1_3487	3398596	3398105	rev.	3398626	+30	**ReviewWeak**
X. bovienii	XBJ1_3802	3682635	3683657	fwd.	3682593	+42	**ReviewWeak**
X. bovienii	XBJ1_3913	3789753	3788374	rev.	3789783	+30	**Change**
X. bovienii	XBJ1_4158	3995207	3992748	rev.	3995348	+141	**Change**
X. bovienii	XBJ1_4169	4006207	4005245	rev.	4006324	+117	**Change**
X. bovienii	XBJ1_4412	4224742	4224416	rev.	4224775	+33	**Change**

C Comparison of Kraken with LMAT

As LMAT requires over 4,300 CPU hours to build its database [84], over half a terabyte of RAM, and superuser privileges to use its memory allocation procedures, we were unable to create a local installation and run LMAT against the three metagenomes we developed to evaluate Kraken. Instead, we have run Kraken against some of the data used in LMAT's published results, and report our comparisons here.

LMAT has the ability to be run with two databases: kFull, the complete database, and kML, a database containing only "marker" k-mers that are the most "taxonomically informative" k-mers in the complete database. kFull (using k = 20) is 619 GB in size, while kML (using k = 18) is 39 GB. The authors reported results using both databases. As the relationship between LMAT-kFull and LMAT-kML is somewhat analogous to that between Kraken and MiniKraken, we report both Kraken and MiniKraken in our comparisons to LMAT.

The PhymmBL dataset used in evaluating LMAT's accuracy was formed by extracting 50 simulated 100 bp reads from each replicon that existed in RefSeq's set of completed bacterial and archaeal genomes as of October 2008 [22]. The simBA-5 metagenome is similar to this dataset, but the addition of simulated error is a crucial difference, as the PhymmBL dataset was simulated without error. Because both LMAT and Kraken used RefSeq genomes to build their k-mer database, and the sequences in the PhymmBL dataset were drawn without modification from that library, both classifiers should achieve very high accuracy because they are being tested on data contained in their training sets. Nonetheless,

as it was the only dataset to which we have access and for which we have LMAT's classification results, we use it here, and report LMAT's published results on this dataset.

The first experiment was to classify the dataset, which contained 540 distinct species, and report the number of species correctly identified in the dataset. Kraken identified 538 species, and MiniKraken identified 536, with neither mistakenly declaring the presence of a species not in the dataset. LMAT-kFull and LMAT-kML also had no false declarations of species presence, but only identified 531 and 527 species, respectively.

In the second experiment, we examined the individual reads to determine if they were classified at the species level, and if so, if they were correctly classified. Neither Kraken nor MiniKraken erroneously classified the species of any read. Kraken correctly identified the species of 88.68% of the reads, and MiniKraken correctly identified 85.06% of the reads' species. LMAT-kFull correctly identified the species of 74.2% of the reads, with 99.8% of species-level classifications being correct. LMAT-kML correctly identified the species of 40.4% of the reads, with 99.7% of species-level classifications being correct.

Since we do not have a local installation of LMAT, we cannot report LMAT's speed on any of our metagenomes. However, LMAT has published speeds for a human microbiome metagenome (SRA ERR011121), consisting of 33,123,975 75 bp reads. LMAT's raw speeds are reported in Kbp/s, but are the result of 40-threaded execution; we therefore divide their reported speeds by 40 here. LMAT-

kFull classified the sample at a speed of 63.7 Kbp/s on a single core, and LMAT-kML classified the sample at a speed of 327.4 Kbp/s on a single core. We downloaded this sample and classified it using Kraken and MiniKraken, using a single thread. Kraken classified the sample in 1005 seconds, for a classification speed of 2473 Kbp/s; MiniKraken took 915 seconds, for a speed of 2714 Kbp/s.

References

1. Glenn TC: **Field guide to next-generation DNA sequencers**. *Mol Ecol Resour* 2011, **11**:759–769.

2. Salzberg S, Delcher A, Kasif S, White O: **Microbial gene identification using interpolated Markov models**. *Nucleic Acids Res* 1998, **26**:544–548.

3. Delcher A, Harmon D, Kasif S, White O, Salzberg S: **Improved microbial gene identification with GLIMMER**. *Nucleic Acids Res* 1999, **27**:4636–4641.

4. Delcher AL, Bratke K, Powers E, Salzberg SL: **Identifying bacterial genes and endosymbiont DNA with Glimmer**. *Bioinformatics* 2007, **23**:673–679.

5. Hyatt D, Chen G-L, Locascio P, Land M, Larimer F, Hauser L: **Prodigal: prokaryotic gene recognition and translation initiation site identification.** *BMC Bioinformatics* 2010, **11**:119.

6. Needleman S, Wunsch C: **A general method applicable to the search for similarities in the amino acid sequence of two proteins**. *J Mol Biol* 1970, **48**:443–453.

7. Smith TF, Waterman MS: **Identification of common molecular subsequences**. *J Mol Biol* 1981, **147**:195–197.

8. Altschul S, Gish W, Miller W, Myers E, Lipman D: **Basic Local Alignment Search Tool**. *J Mol Biol* 1990, **215**:403–410.

9. Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, Madden T: **BLAST+: architecture and applications**. *BMC Bioinformatics* 2009, **10**:421.

10. Kent J: **BLAT—The BLAST-Like Alignment Tool**. *Genome Res* 2002, **12**:656–664.

11. Li R, Li Y, Kristiansen K, Wang J: **SOAP: short oligonucleotide alignment program**. *Bioinformatics* 2008, **24**:713–714.

12. Li H, Ruan J, Durbin R: **Mapping short DNA sequencing reads and calling variants using mapping quality scores**. *Genome Res* 2008, **18** (11):1851–1858.

13. Langmead B, Trapnell C, Pop M, Salzberg S: **Ultrafast and memory-efficient alignment of short DNA sequences to the human genome**. *Genome Biol* 2009, **10**:R25–10.

14. Langmead B, Salzberg S: **Fast gapped-read alignment with Bowtie 2**. *Nat Methods* 2012, **9**:357–359.

15. Li H, Durbin R: **Fast and accurate short read alignment with Burrows–Wheeler transform**. *Bioinformatics* 2009, **25**:1754–1760.

16. Burrows M, Wheeler DJ: *A Block-Sorting Lossless Data Compression Algorithm.* 1994.

17. Ferragina P, Manzini G: **Opportunistic data structures with applications**. *Found Comput Sci 2000 Proceedings 41st Annu Symp* 2000:390–398.

18. Thompson JD, Higgins DG, Gibson TJ: **CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice**. *Nucleic Acids Res* 1994, **22** (22):4673–4680.

19. Edgar R: **MUSCLE: multiple sequence alignment with high accuracy and high throughput.** *Nucleic Acids Res* 2004, **32**:1792–1797.

20. Alföldi J, Lindblad-Toh K: **Comparative genomics as a tool to understand evolution and disease**. *Genome Res* 2013, **23** (7):1063–1068.

21. Hardison RC: **Comparative Genomics**. *PLoS Biol* 2003, **1**:e58.

22. Brady A, Salzberg SL: **Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models.** *Nat Methods* 2009, **6**:673–676.

23. Wood DE, Salzberg SL: **Kraken: ultrafast metagenomic sequence classification using exact alignments**. *Genome Biol* 2014, **15**:R46.

24. Turnbaugh PJ, Hamady M, Yatsunenko T, Cantarel BL, Duncan A, Ley RE, Sogin ML, Jones WJ, Roe BA, Affourtit JP, Egholm M, Henrissat B, Heath AC, Knight R, Gordon JI: **A core gut microbiome in obese and lean twins**. *Nature* 2009, **457**:480–484.

25. Tyson G, Chapman J, Hugenholtz P, Allen E, Ram R, Richardson P, Solovyev V, Rubin E, Rokhsar D, Banfield J: **Community structure and metabolism through reconstruction of microbial genomes from the environment.** *Nature* 2004, **428**:37–43.

26. Fleischmann RD, Adams MD, White O, Clayton RA, Kirkness EF, Kerlavage AR, Bult CJ, Tomb JF, Dougherty BA, Merrick JM, Al E: **Whole-genome random sequencing and assembly of Haemophilus influenzae Rd**. *Science* 1995, **269**:496–512.

27. Lagesen K, Ussery DW, Wassenaar TM: **Genome update: the 1000th genome – a cautionary tale**. *Microbiol* 2010, **156** (3):603–608.

28. Pruitt K, Tatusova T, Maglott D: **NCBI reference sequences (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins.** *Nucleic Acids Res* 2007, **35**(Database issue):D61–D65.

29. Pruitt KD, Tatusova T, Brown GR, Maglott DR: **NCBI Reference Sequences (RefSeq): current status, new features and genome annotation policy.** *Nucleic Acids Res* 2012, **40**(Database issue):D130–5.

30. Poptsova M, Gogarten P: **Using comparative genome analysis to identify problems in annotated microbial genomes**. *Microbiology* 2010, **156**:1909–1917.

31. Suzek B, Ermolaeva M, Schreiber M, Salzberg S: **A probabilistic method for identifying start codons in bacterial genomes**. *Bioinformatics* 2001, **17**:1123–1130.

32. Huttenhower C, Gevers D, Knight R, Abubucker S, Badger JH, Chinwalla AT, Creasy HH, Earl AM, FitzGerald MG, Fulton RS, Giglio MG, Hallsworth-Pepin K, Lobos EA, Madupu R, Magrini V, Martin JC, Mitreva M, Muzny DM, Sodergren EJ, Versalovic J, Wollam AM, Worley KC, Wortman JR, Young SK, Zeng Q, Aagaard KM, Abolude OO, Allen-Vercoe E, Alm EJ, Alvarado L, et al.: **Structure, function and diversity of the healthy human microbiome**. *Nature* 2012, **486**:207–214.

33. Segata N, Waldron L, Ballarini A, Narasimhan V, Jousson O, Huttenhower C: **Metagenomic microbial community profiling using unique clade-specific marker genes**. *Nat Methods* 2012, **9**:811–814.

34. Liu B, Gibbons T, Ghodsi M, Treangen T, Pop M: **Accurate and fast estimation of taxonomic profiles from metagenomic shotgun sequences**. *BMC Genomics* 2011, **12**(Suppl 2):S4.

35. Roberts M, Hayes W, Hunt B, Mount S, Yorke J: **Reducing storage requirements for biological sequence comparison**. *Bioinformatics* 2004, **20**:3363–3369.

36. Wood D, Lin H, Levy-Moonshine A, Swaminathan R, Chang Y-C, Anton B, Osmani L, Steffen M, Kasif S, Salzberg S: **Thousands of missed genes found in bacterial genomes and their analysis with COMBREX**. *Biol Direct* 2012, **7**:37.

37. Lukashin A, Borodovsky M: **GeneMark.hmm: New solutions for gene finding**. *Nucleic Acids Res* 1998, **26**:1107–1115.

38. Besemer J, Lomsadze A, Borodovsky M: **GeneMarkS: a self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions**. *Nucleic Acids Res* 2001, **29**:2607–2618.

39. Skovgaard M, Jensen LJ, Brunak S, Ussery D, Krogh A: **On the total number of genes and their length distribution in complete microbial genomes.** *Trends Genet* 2001, **17**:425–8.

40. Benson D, Karsch-Mizrachi I, Lipman D, Ostell J, Sayers E: **GenBank**. *Nucleic Acids Res* 2011, **39**(Database issue):D32–D37.

41. Roberts R, Chang Y-C, Hu Z, Rachlin J, Anton B, Pokrzywa R, Choi H-P, Faller L, Guleria J, Housman G, Klitgord N, Mazumdar V, McGettrick M, Osmani L, Swaminathan R, Tao K, Letovsky S, Vitkup D, Segrè D, Salzberg S, Delisi C, Steffen M, Kasif S: **COMBREX: a project to accelerate the functional annotation of prokaryotic genomes**. *Nucleic Acids Res* 2011, **39**(suppl 1):D11–D14.

42. **COMBREX Website** [http://combrex.bu.edu/]

43. Warren A, Archuleta J, Feng W, Setubal J: **Missing genes in the annotation of prokaryotic genomes**. *BMC Bioinformatics* 2010, **11**:131.

44. **COMBREX Missed Genes** [http://combrex.bu.edu/missedGenes]

45. Kerfeld CA, Scott KM: **Using BLAST to teach "E-value-tionary" concepts.** *PLoS Biol* 2011, **9**:e1001014.

46. Klimke W, Agarwala R, Badretdin A, Chetvernin S, Ciufo S, Fedorov B, Kiryutin B, O'Neill K, Resch W, Resenchuk S, Schafer S, Tolstoy I, Tatusova T: **The National Center for Biotechnology Information's Protein Clusters Database**. *Nucleic Acids Res* 2009, **37**(suppl 1):D216–D223.

47. Liu B, Pop M: **ARDB—Antibiotic Resistance Genes Database**. *Nucleic Acids Res* 2009, **37**(suppl 1):D443–D447.

48. Liu A, Tran L, Becket E, Lee K, Chinn L, Park E, Tran K, Miller J: **Antibiotic Sensitivity Profiles Determined with an Escherichia coli Gene Knockout Collection: Generating an Antibiotic Bar Code**. *Antimicrob Agents Chemother* 2010, **54**:1393–1403.

49. Eberhardt RY, Haft DH, Punta M, Martin M, O'Donovan C, Bateman A: **AntiFam: a tool to help identify spurious ORFs in protein annotation.** *Database* 2012, **2012**:bas003.

50. Ecker D, Sampath R, Willett P, Wyatt J, Samant V, Massire C, Hall T, Hari K, McNeil J, Büchen-Osmond C, Budowle B: **The Microbial Rosetta Stone Database: A compilation of global and emerging infectious microorganisms and bioterrorist threat agents**. *BMC Microbiol* 2005, **5**:19.

51. Jiménez E, Langa S, Martín V, Arroyo R, Martín R, Fernández L, Rodríguez JM: **Complete genome sequence of Lactobacillus fermentum CECT 5716, a probiotic strain isolated from human milk.** *J Bacteriol* 2010, **192**:4800.

52. Lewis, L.A., Gillaspy, A.F., McLaughlin, R.E., Gipson, M., Ducey TF, Ownbey, T., Hartman, K., Nydick, C., Carson, M.B., Vaughn J, Thomson, C., Song, L., Lin, S., Yuan, X., Najar, F., Zhan, M., Ren Q, Zhu, H., Qi, S., Kenton, S.M., Lai, H., White, J.D., Clifton S, Roe, B.A. and Dyer DW: **Neisseria gonorrhoeae FA 1090**. 2010:GenBank accession no. AE004969.

53. Donati, C., Hiller, N.L., Tettelin, H., Muzzi, A., Croucher NJ, Angiuoli, S.V., Oggioni, M., Dunning Hotopp, J.C., Hu, F.Z., Riley D, Covacci, A., Mitchell, T.J., Bentley, S.D., Kilian, M., Ehrlich GD, Rappuoli, R., Moxon, E.R. and Masignani V: **Streptococcus pneumoniae INV104**. 2011:GenBank accession no. FQ312030.

54. Shen X, Wang Q, Xia L, Zhu X, Zhang Z, Liang Y, Cai H, Zhang E, Wei J, Chen C, Song Z, Zhang H, Yu D, Hai R: **Complete genome sequences of Yersinia pestis from natural foci in China.** *J Bacteriol* 2010, **192**:3551–3552.

55. Toh H, Weiss B, Perkin S, Yamashita A, Oshima K, Hattori M, Aksoy S: **Massive genome erosion and functional adaptations provide insights into the symbiotic lifestyle of Sodalis glossinidius in the tsetse host.** *Genome Res* 2006, **16**:149–156.

56. Bruggemann H, Baumer S, Fricke WF, Wiezer A, Liesegang H, Decker I, Herzberg C, Martinez-Arias R, Merkl R, Henne A, Gottschalk G: **The genome sequence of Clostridium tetani, the causative agent of tetanus disease.** *Proc Natl Acad Sci U S A* 2003, **100**:1316–1321.

57. Challacombe J, Altherr M, Xie G, Bhotika S, Brown N, Bruce D, Campbell C, Campbell M, Chen J, Chertkov O, Cleland C, Dimitrijevic M, Doggett N, Fawcett J, Glavina T, Goodwin L, Green L, Han C, Hill K, Hitchcock P, Jackson P, Keim P, Ramesh A, Longmire J, Lucas S, Malfatti S, Martinez D, McMurry K, Meincke L, Misra M, et al.: **The complete genome sequence of Bacillus thuringiensis Al Hakam.** *J Bacteriol* 2007, **189**:3680–3681.

58. Sakiyama T, Takami H, Ogasawara N, Kuhara S, Kozuki T, Doga K, Ohyama A, Horikoshi K: **An automated system for genome analysis to support microbial whole-genome shotgun sequencing.** *Biosci Biotechnol Biochem* 2000, **64**:670–673.

59. Johnson T, Kariyawasam S, Wannemuehler Y, Mangiamele P, Johnson S, Doetkott C, Skyberg J, Lynne A, Johnson J, Nolan L: **The Genome Sequence of Avian Pathogenic Escherichia coli Strain O1:K1:H7 Shares Strong Similarities with Human Extraintestinal Pathogenic E. coli Genomes**. *J Bacteriol* 2007, **189**:3228–3236.

60. Ou H, Guo F, Zhang C: **GS-Finder: a program to find bacterial gene start sites with a self-training method**. *Int J Biochem Cell Biol* 2004, **36**:535–544.

61. Tech M, Morgenstern B, Meinicke P: **TICO: a tool for postprocessing the predictions of prokaryotic translation initiation sites**. *Nucleic Acids Res* 2006, **34**(suppl 2):W588–W590.

62. Pop M, Phillippy A, Delcher A, Salzberg S: **Comparative genome assembly**. *Brief Bioinform* 2004, **5**:237–248.

63. Frishman D, Mironov A, Mewes H-W, Gelfand M: **Combining diverse evidence for gene recognition in completely sequenced bacterial genomes**. *Nucleic Acids Res* 1998, **26**:2941–2947.

64. Badger JH, Olsen GJ: **CRITICA: coding region identification tool invoking comparative analysis.** *Mol Biol Evol* 1999, **16**:512–524.

65. Walker M, Pavlovic V, Kasif S: **A comparative genomic method for computational identification of prokaryotic translation initiation sites**. *Nucleic Acids Res* 2002, **30**:3181–3191.

66. Wall M, Raghavan S, Cohn J, Dunbar J: **Genome majority vote improves gene predictions.** *PLoS Comput Biol* 2011, **7**:e1002284.

67. Pertea M, Ayanbule K, Smedinghoff M, Salzberg S: **OperonDB: a comprehensive database of predicted operons in microbial genomes**. *Nucleic Acids Res* 2009, **37**(suppl 1):D479–D482.

68. Henikoff S, Henikoff J: **Amino acid substitution matrices from protein blocks.** *Proc Natl Acad Sci U S A* 1992, **89**:10915–10919.

69. Rudd K: **EcoGene: a genome sequence database for Escherichia coli K-12.** *Nucleic Acids Res* 2000, **28**:60–64.

70. Aivaliotis M, Gevaert K, Falb M, Tebbe A, Konstantinidis K, Bisle B, Klein C, Martens L, Staes A, Timmerman E, Van Damme J, Siedler F, Pfeiffer F, Vandekerckhove J, Oesterhelt D: **Large-Scale Identification of N-Terminal Peptides in the Halophilic Archaea Halobacterium salinarum and Natronomonas pharaonis**. *J Proteome Res* 2007, **6**:2195–2204.

71. Su'etsugu M, Nakamura K, Keyamura K, Kudo Y, Katayama T: **Hda Monomerization by ADP Binding Promotes Replicase Clamp-mediated DnaA-ATP Hydrolysis**. *J Biol Chem* 2008, **283**:36118–36131.

72. Brombach M, Pon CL: **The unusual translational initiation codon AUU limits the expression of the infC (initiation factor IF3) gene of Escherichia coli.** *Mol Gen Genet* 1987, **208**:94–100.

73. Brazilian National Genome Project Consortium: **The complete genome sequence of Chromobacterium violaceum reveals remarkable and exploitable bacterial adaptability.** *Proc Natl Acad Sci U S A* 2003, **100**:11660–11665.

74. Slater S, Goldman B, Goodner B, Setubal J, Farrand S, Nester E, Burr T, Banta L, Dickerman A, Paulsen I, Otten L, Suen G, Welch R, Almeida N, Arnold F, Burton O, Du Z, Ewing A, Godsy E, Heisel S, Houmiel K, Jhaveri J, Lu J, Miller N, Norton S, Chen Q, Phoolcharoen W, Ohlin V, Ondrusek D, Pride N, et al.: **Genome sequences of three agrobacterium biovars help elucidate the evolution of multichromosome genomes in bacteria.** *J Bacteriol* 2009, **191**:2501–2511.

75. Giraud E, Moulin L, Vallenet D, Barbe V, Cytryn E, Avarre J-C, Jaubert M, Simon D, Cartieaux F, Prin Y, Bena G, Hannibal L, Fardoux J, Kojadinovic M, Vuillet L, Lajus A, Cruveiller S, Rouy Z, Mangenot S, Segurens B, Dossat C, Franck W, Chang W-S, Saunders E, Bruce D, Richardson P, Normand P, Dreyfus B, Pignol D, Stacey G, et al.: **Legumes symbioses: absence of Nod genes in photosynthetic bradyrhizobia.** *Science* 2007, **316**:1307–1312.

76. Vasconcelos ATR, Ferreira HB, Bizarro C V, Bonatto SL, Carvalho MO, Pinto PM, Almeida DF, Almeida LGP, Almeida R, Alves-Filho L, Assuncao EN, Azevedo VAC, Bogo MR, Brigido MM, Brocchi M, Burity HA, Camargo AA, Camargo SS, Carepo MS, Carraro DM, de Mattos Cascardo JC, Castro LA, Cavalcanti G, Chemale G, Collevatti RG, Cunha CW, Dallagiovanna B, Dambros BP, Dellagostin OA, Falcao C, et al.: **Swine and Poultry Pathogens: the Complete Genome Sequences of Two Strains of Mycoplasma hyopneumoniae and a Strain of Mycoplasma synoviae**. *J Bacteriol* 2005, **187**:5568–5577.

77. Stephan R, Lehner A, Tischler P, Rattei T: **Complete genome sequence of Cronobacter turicensis LMG 23827, a food-borne pathogen causing deaths in neonates.** *J Bacteriol* 2011, **193**:309–310.

78. Blanc G, Ogata H, Robert C, Audic S, Claverie J-M, Raoult D: **Lateral gene transfer between obligate intracellular bacteria: evidence from the Rickettsia massiliae genome.** *Genome Res* 2007, **17**:1657–1664.

79. Venter C, Remington K, Heidelberg J, Halpern A, Rusch D, Eisen J, Wu D, Paulsen I, Nelson K, Nelson W, Fouts D, Levy S, Knap A, Lomas M, Nealson K,

White O, Peterson J, Hoffman J, Parsons R, Baden-Tillson H, Pfannkoch C, Rogers Y-H, Smith H: **Environmental Genome Shotgun Sequencing of the Sargasso Sea**. *Science* 2004, **304**:66–74.

80. Huson D, Auch A, Qi J, Schuster S: **MEGAN analysis of metagenomic data.** *Genome Res* 2007, **17**:377–386.

81. Brady A, Salzberg S: **PhymmBL expanded: confidence scores, custom databases, parallelization and more**. *Nat Methods* 2011, **8**:367.

82. Rosen G, Garbarine E, Caseiro D, Polikar R, Sokhansanj B: **Metagenome Fragment Classification Using N-Mer Frequency Profiles**. *Adv Bioinformatics* 2008, **2008**:1–12.

83. Treangen T, Koren S, Sommer D, Liu B, Astrovskaya I, Ondov B, Darling A, Phillippy A, Pop M: **MetAMOS: a modular and open source metagenomic assembly and analysis pipeline**. *Genome Biol* 2013, **14**:R2.

84. Ames SK, Hysom DA, Gardner SN, Lloyd GS, Gokhale MB, Allen JE: **Scalable metagenomic taxonomy classification using a reference genome database**. *Bioinformatics* 2013, **29**:2253–2260.

85. Kindblom C, Davies JR, Herzberg MC, Svensäter G, Wickström C: **Salivary proteins promote proteolytic activity in Streptococcus mitis biovar 2 and Streptococcus mutans**. *Mol Oral Microbiol* 2012, **27**:362–372.

86. Foweraker JE, Cooke NJ, Hawkey PM: **Ecology of Haemophilus influenzae and Haemophilus parainfluenzae in sputum and saliva and effects of antibiotics on their distribution in patients with lower respiratory tract infections.** *Antimicrob Agents Chemother* 1993, **37**:804–809.

87. Könönen E, Saarela M, Karjalainen J, Jousimies-Somer H, Alaluusua S, Asikainen S: **Transmission of oral Prevotella melaninogenica between a mother and her young child.** *Oral Microbiol Immunol* 1994, **9**:310–314.

88. Zimin A V., Marçais G, Puiu D, Roberts M, Salzberg SL, Yorke JA: **The MaSuRCA genome assembler**. *Bioinformatics* 2013, **29**:2669–2677.

89. Marçais G, Kingsford C: **A fast, lock-free approach for efficient parallel counting of occurrences of k-mers**. *Bioinformatics* 2011, **27**:764–770.

90. Magoc T, Pabinger S, Canzar S, Liu X, Su Q, Puiu D, Tallon LJ, Salzberg SL: **GAGE-B: an evaluation of genome assemblers for bacterial organisms**. *Bioinformatics* 2013, **29**:1718–1725.

91. Mavromatis K, Ivanova N, Barry K, Shapiro H, Goltsman E, McHardy A, Rigoutsos I, Salamov A, Korzeniewski F, Land M, Lapidus A, Grigoriev I, Richardson P, Hugenholtz P, Kyrpides N: **Use of simulated data sets to evaluate the fidelity of metagenomic processing methods**. *Nat Methods* 2007, **4**:495–500.

92. Ondov B, Bergman N, Phillippy A: **Interactive metagenomic visualization in a Web browser**. *BMC Bioinformatics* 2011, **12**:385.

93. Salzberg SL, Pertea M, Fahrner JA, Sobreira N: **DIAMUND: Direct Comparison of Genomes to Detect Mutations**. *Hum Mutat* 2014, **35**:283–288.