Comparative Analysis of Genomic Similarity Tools in Species Identification

Chandra Sekhar Nerella

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science & Applications

Lenwood S. Heath, Co-chair
Boris A. Vinatzer, Co-chair
Anuj Karpatne

December 16, 2024
Blacksburg, Virginia

Keywords: Genomic Similarity, Genome Comparison Tools, Average Nucleotide Identity (ANI), Alignment-Based Tools, Alignment-Free Tools

Comparative Analysis of Genomic Similarity Tools in Species Identification

Chandra Sekhar Nerella

ABSTRACT

This study presents the development and evaluation of an automated pipeline for genome comparison, leveraging four bioinformatics tools: alignment-based methods (pyANI, FastANI) and k-mer-based methods (Sourmash, BinDash 2.0). The analysis focuses on high-quality genomic datasets characterized by 100% completeness, ensuring consistency and accuracy in the comparison process. The pipeline processes genomes under uniform conditions, recording key performance metrics such as execution time and rank correlations. Initial comparisons were conducted on a subset of five genomes, generating 10 unique pairwise comparisons to establish baseline performance. This preliminary analysis identified $k = 10$ as the optimal k-mer size for Sourmash and BinDash, significantly improving their comparability with alignment-based methods.

For the expanded dataset of 175 genomes, encompassing $\binom{175}{2}$=15,225 unique comparisons, pyANI and FastANI demonstrated high similarity values, often exceeding 90% for closely related genomes. Rank correlations, calculated using Spearman's ρ and Kendall's τ, highlighted strong agreement between pyANI and FastANI ($\rho = 0.9630$, $\tau = 0.8625$) due to their shared alignment-based methodology. Similarly, Sourmash and BinDash, both employing k-mer-based approaches, exhibited moderate-to-strong rank correlations ($\rho = 0.6967$, $\tau = 0.5290$). In contrast, the rank correlations between alignment-based and k-mer-based tools were lower, underscoring methodological differences in genome similarity calculations.

Execution times revealed significant contrasts between the tools. Alignment-based meth-

ods required substantial computation time, with pyANI taking an average of 1.97 seconds per comparison and FastANI averaging 0.81 seconds per comparison. Conversely, k-mer-based methods demonstrated exceptional computational efficiency, with Sourmash completing comparisons in 2.1 milliseconds and BinDash in just 0.25 milliseconds per comparison, reflecting a difference of nearly three orders of magnitude between the two categories. These results underscore the trade-offs between computational cost and methodological approaches in genome similarity estimation.

This study provides valuable insights into the relative strengths and weaknesses of genome comparison tools, offering a comprehensive framework for selecting appropriate methods for diverse genomic research applications. The findings emphasize the importance of parameter optimization for k-mer-based tools and highlight the scalability of these methods for large-scale genomic analyses.

Comparative Analysis of Genomic Similarity Tools in Species Identification

Chandra Sekhar Nerella

GENERAL AUDIENCE ABSTRACT

This study explores the strengths and weaknesses of different tools used to compare genomes, which are the complete set of DNA in living organisms. Comparing genomes allows scientists to understand how different species are related, uncover shared traits, and identify what makes each species unique. The tools we examined fall into two main categories: detailed tools (called alignment-based methods) and faster, more approximate tools (called k-mer-based methods). The detailed tools, such as pyANI and FastANI, compare DNA sequences piece by piece, providing very accurate results. In contrast, the faster tools, such as Sourmash and BinDash, look for patterns in smaller sections of DNA, which makes them much quicker but sometimes less precise.

To start, we tested these tools on a small group of genomes to see how they performed. By adjusting a setting in the faster tools, we found that their results became more similar to the detailed tools, improving their reliability. Encouraged by these findings, we expanded the comparison to a much larger dataset of 175 genomes. For this larger dataset, the detailed tools provided highly accurate results but required much more time and computational power. On the other hand, the faster tools completed the comparisons in a fraction of the time, making them ideal for larger datasets where quick results are needed.

We also compared how the tools ranked genome similarities and found that tools using similar methods, like pyANI and FastANI, had very consistent rankings. Likewise, the faster tools, Sourmash and BinDash, also agreed with each other. However, the rankings between the two

types of tools (detailed versus faster) were less consistent, reflecting their different approaches to genome comparison.

This research provides a practical guide for scientists choosing tools to compare genomes. If accuracy and detail are most important, alignment-based tools are the best choice, though they take more time and computational resources. If speed is critical, such as when working with very large datasets, k-mer-based tools offer an excellent alternative. By understanding the strengths and trade-offs of each method, researchers can make informed decisions to suit their specific needs, whether focusing on small, detailed studies or large-scale genome analyses.

Dedicated to my family.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Professor Lenwood Heath, who guided me throughout the entirety of my thesis. His mentorship was instrumental in shaping my research journey. One of the primary reasons I pursued a Master's degree was my love for research, but in the beginning, I struggled to define the problem or idea I wanted to explore. Professor Heath helped me brainstorm ideas and refine my direction with incredible patience and wisdom. I am especially grateful to Reza Mazloom, Jingyi (Eve) Zhang, Yoonjim Kim, Sahar Desai, Atul Bharadwaj, and Kassaye Belay, who participated in the brainstorming sessions and pitched valuable ideas that eventually led to this work. Their support and insights during this early phase made a lasting impact.

I would also like to extend my heartfelt thanks to my co-advisor, Professor Boris Vinatzer, whose role in this thesis was crucial from the very beginning. Despite his demanding schedule, he was always available to answer my questions and offer guidance. His enthusiasm for the genomeRxiv project we collaborated on, along with his meticulous attention to detail, inspired me to stay focused and work diligently. His mentorship helped me grow not only as a researcher but also as a problem-solver with a broader perspective.

My gratitude extends to Professor Anuj Karpatne, whose introductory Machine Learning course (CS 5805) was the first Machine Learning course I took. I instantly knew I wanted him on my committee. His passion for the subject and dedication to teaching were contagious and left an indelible mark on me as a student. His course not only increased my interest in Machine Learning but also instilled in me a commitment to work hard and approach any task with sincerity. This inspiration carried over to every aspect of my research.

I want to give a special mention to Reza Mazloom, who was incredibly supportive throughout

my time as a Master's student. Whether it was related to my thesis or the genomeRxiv project, he was always there to help. Reza's advice went beyond technical assistance; he taught me how to think critically and adopt practices that are essential for becoming a successful software engineer. The lessons I learned from him will stay with me throughout my career, and I will always be grateful for his guidance.

Once again, I would like to express my sincere thanks to Professor Lenwood Heath. Even though I had to complete my thesis in a relatively short time frame, he consistently met with me, providing patient guidance. From helping with the literature review to identifying the right tools and resources, his support was invaluable in helping me make significant progress. His dedication and mentorship have been a cornerstone of my academic journey.

Finally, I would like to thank my family and especially my dad. Their unwavering support and encouragement have been the foundation that has allowed me to pursue my dreams. Without them, none of this would have been possible.

Contents

List of Figures

List of Tables

List of Abbreviations

GTDB Genome Taxonomy Database

NCBI National Center for Biotechnology Information

NLP Natural Language Processing

Chapter 1

Introduction

In the quest to understand the intricacies of life at a molecular level, genomic comparisons have emerged as a cornerstone of modern biological research. These comparisons provide the framework for deciphering the complex genetic blueprints that define every living organism. From unraveling the mysteries of evolutionary biology to advancing personalized medicine, the ability to compare genomic sequences across different species and populations has transformed our approach to tackling some of the most pressing challenges in healthcare, agriculture, and environmental conservation. As we delve deeper into the genomic era, the importance of refining the tools and techniques for genome analysis cannot be overstated, with each advancement paving the way for new discoveries that could reshape our understanding of biology and beyond.

As the volume of genomic data continues to grow exponentially, the need for efficient and accurate genome-similarity comparison tools has become increasingly critical. Traditional methods of genome comparison, such as alignment-based approaches, while highly accurate, struggle with the computational demands of large-scale genomic analyses [1]. This has led to the development of a new generation of tools that employ innovative algorithms and data structures to rapidly assess genomic similarity.

The rapid increase in nucleic acid data has led to the creation of several international databases designed to organize this vast amount of information. As molecular biology continues to evolve, the complexity of relationships between sequences and the number of features

of interest have also increased. This growing complexity necessitates the development of advanced methods for sequence comparison, which is one of the most mathematically developed aspects of macromolecular sequence analysis [2].

The Basic Local Alignment Search Tool (BLAST) has been pivotal in genomic analysis since its 1990 introduction [3], offering a fast, heuristic approach to identifying local sequence similarities [3]. By comparing nucleotide or protein sequences to databases and calculating statistical significance, BLAST has revolutionized sequence comparison. Its impact extends beyond direct applications, inspiring tools like sourmash and pyANI, which build on BLAST's principles to tackle large-scale genomic comparisons. BLAST incorporates dynamic programming in its alignment extension phase, enhancing both speed and sensitivity by extending initial high-scoring matches into longer alignments [4]. This integration allows BLAST to maintain accuracy while reducing computational time compared to exhaustive dynamic programming methods [4]. The foundation of alignment-based algorithms, including BLAST, lies in assigning scores to evolutionary events such as insertions, deletions, and substitutions. These algorithms aim to compute optimal alignments that minimize mutation costs, a principle rooted in the work of Needleman and Wunsch (1970), forming the basis for quantifying sequence similarity and reconstructing evolutionary relationships [5].

Building upon the foundation laid by BLAST, researchers have developed more specialized tools for comparing whole genomes, with average nucleotide identity (ANI) emerging as a key metric for assessing genomic similarity between prokaryotic organisms. ANI extends the concept of sequence similarity that BLAST pioneered to a genome-wide scale, providing a robust measure of genetic relatedness between bacterial and archaeal species. The original method for computing ANI, often referred to as ANIb, utilizes BLAST as its core alignment algorithm [6].

ANI values around 95-96 percentage range have been widely adopted as a threshold for

species boundaries in prokaryotes [7]. This threshold correlates well with the traditional 70 percent DNA-DNA hybridization standard used for decades in bacterial taxonomy [8]. There are several genome-similarity comparison tools that have incorporated ANI as a key metric. FastANI is a rapid alignment-free tool that approximates ANI using a MinHash-based approach, allowing for quick comparisons of large genomic data sets [9]. Similarly pyANI is a python module that implements various ANI methods, including ANIb, ANIm, and alignment-free approaches such as TETRA, providing flexibility for different research needs. OrthoANI ia an improved ANI algorithm that focuses on orthologous gene pairs, potentially providing more accurate species delineation [10].

These tools have enabled researchers to perform large-scale genomic comparisons efficiently, contributing to our understanding of microbial diversity and evolution. The widespread adoption of ANI in genomic studies underscores its value as a robust and meaningful measure of prokaryotic genome similarity. However, it is important to note that while ANI provides a powerful measure for species delineation, it should be used in conjunction with other taxonomic approaches as part of a polyphasic taxonomy to ensure accurate and comprehensive classification of prokaryotic organisms [11][7].

As the field of genomic analysis continues to evolve, a diverse array of genome-similarity comparison tools has emerged, each with its own strengths and limitations. This proliferation of tools presents both opportunities and challenges for researchers. While the abundance of options allows for tailored approaches to specific research questions, it also necessitates a careful evaluation of each tool's performance characteristics.

The selection of an appropriate genome-similarity comparison tool often involves considering various trade-offs. For instance, some tools prioritize speed and computational efficiency, making them suitable for large-scale analyses but potentially sacrificing some degree of accuracy. Others may offer higher precision but at the cost of increased computational time

and resource requirements. Factors such as the size of the data set, the degree of similarity between the genomes being compared, and the specific research objectives all play crucial roles in determining the most suitable tool for a given study.

Moreover, the rapid pace of technological advancement in genomics continually shifts the landscape of these tools. New algorithms and computational techniques are regularly introduced, promising improvements in speed, accuracy, or both. This dynamic environment underscores the need for ongoing, comprehensive evaluations of genome-similarity comparison tools.

Given this context, there is a pressing need for systematic comparisons of these tools across a range of performance metrics. Such comparisons can provide valuable insights into the relative strengths and weaknesses of each tool, guiding researchers in their selection process. Beyond traditional concerns of accuracy and error rates, key factors in these evaluations include the time required by each tool to perform genome comparisons and how well the similarity percentages align across tools. These aspects not only reflect computational efficiency but also reveal the consistency in relative rankings of genome pairs, providing a more holistic understanding of tool performance.

This thesis aims to address this need by conducting a comprehensive comparison of several prominent genome-similarity comparison tools. By evaluating these tools across multiple performance dimensions, we seek to provide a nuanced understanding of their capabilities and limitations. The goal is to provide a comprehensive analysis that will not only assist researchers in choosing the most appropriate tool for their specific needs but also contribute to the ongoing refinement and development of genome comparison methodologies.

In light of the diverse landscape of genome-similarity comparison tools and the need for comprehensive evaluation, this thesis will focus on a comparative analysis of four prominent

tools: Sourmash[12], PyANI[13], FastANI[9], and Bindash 2.0[14].

These tools represent a range of approaches to genome comparison, from alignment-free methods to traditional alignment-based techniques, and from rapid approximation algorithms to more precise calculations. Sourmash utilizes MinHash sketching for quick genome comparisons, ANI calculation methods and for the purpose of this paper, pyANIm and pyANIb will be compared with the other tools, FastANI offers a rapid alignment-free ANI estimation, and Bindash 2.0 employs advanced binary sketching techniques. By thoroughly examining these tools across multiple performance dimensions, including accuracy, speed, scalability, and applicability to various genomic data sets, this study aims to guide researchers with their choice of genome comparison methods and contribute to the ongoing advancement of comparative genomics.

Chapter 2

Tools

2.1 FastANI

FastANI (Fast Average Nucleotide Identity) is a tool developed for rapid, alignment-free computation of whole-genome Average Nucleotide Identity (ANI). ANI is defined as the mean nucleotide identity of orthologous gene pairs shared between two microbial genomes . FastANI was designed to efficiently process a large number of genome assemblies with modest computational resources, addressing the need for quick genomic comparisons in the era of big data genomics [9][15].

Unlike traditional ANI calculation methods, FastANI avoids expensive sequence alignments. Instead, it uses Mashmap as its MinHash-based sequence mapping engine to compute orthologous mappings and alignment identity estimates. Because of this, FastANI achieves two to three orders of magnitude speedup compared to BLAST-based ANI solvers. In various data sets, FastANI demonstrated runtime improvements ranging from 50x to 4608x [9]. Despite its speed, FastANI maintains accuracy comparable to alignment-based methods, especially for genomes with more than 80 percent similarity. FastANI can be parallelized easily, allowing for even faster processing of large data sets. It has been shown to scale efficiently with up to 80 parallel processes [9]. FastANI includes a feature to visualize conserved regions between two genomes, providing insights into evolutionary relationships [9][1]

The way FastANI works is something worth noting. FastANI first fragments the query genome into smaller segments. These segments are then mapped to the reference genome using MashMap, which identifies the best orthologous matches. The ANI is computed based on the mean identity of these reciprocal best matches. This process is performed in both directions (query to reference and reference to query) to ensure accuracy by focusing on orthologous genes and excluding paralogs [9].

FastANI's innovative use of MinHash-based mapping allows it to bypass the alignment step, resulting in a tool that is both fast and accurate, making it an invaluable resource for researchers conducting large-scale genomic analyses. Its ability to quickly process large numbers of genome pairs makes it particularly useful for delineating species boundaries in microbial genomics [9].

In conclusion, FastANI represents a significant advancement in genomic comparison tools, offering a balance of speed, accuracy, and scalability that makes it well-suited for the demands of modern genomic research. Its ability to rapidly process large numbers of genome pairs while maintaining accuracy comparable to traditional methods has made it an invaluable tool for large-scale comparative genomics studies and bacterial species delineation. As genomic databases continue to grow, tools like FastANI will play an increasingly crucial role in our understanding of microbial diversity and evolution.

2.2 PyANI

PyANI (Python Average Nucleotide Identity) is a Python module and standalone program designed for calculating whole-genome similarity measures, particularly Average Nucleotide Identity (ANI). PyANI implements several methods for ANI calculation, including ANIb (ANI using BLAST+), ANIblastall (ANI using legacy BLAST), ANIm (ANI using MUM-

mer), fastANI (ANI using the fastANI algorithm), TETRA (tetranucleotide frequency correlation).

Clearly different versions of PyANI implement different methods. For example ANIb and ANIblastall use BLAST+ or legacy BLAST to align genome fragments, ANIm utilizes MUMmer for whole-genome alignment and the one that implements fastANI implements the fastANI algorithm for rapid, alignment-free ANI estimation. PyANI calculates the nucleotide identity for each aligned region or matched segment. The final ANI value is determined by averaging the identity scores across all comparisons. PyANI takes advantage of multicore systems to parallelize computations, significantly speeding up the analysis for large data sets. Another main advantage of using PyANI is that it provides various output formats, including graphical representations of ANI results, making it easier to interpret genomic relationships.

In summary, PyANI stands out as a versatile and comprehensive tool for genomic comparison, offering researchers a range of methods for calculating Average Nucleotide Identity. Its flexibility in implementing multiple ANI algorithms, coupled with its ability to handle large data sets through parallelization, makes it a valuable asset in comparative genomics. The integration of various output formats, including graphical representations, enhances the interpretability of results. As genomic databases continue to expand, PyANI's adaptability and efficiency position it as an essential tool for researchers seeking to unravel the complexities of microbial genomic relationships and contribute to our understanding of prokaryotic diversity and evolution.

2.3 Sourmash

Sourmash is a powerful and versatile tool designed for rapid genome and metagenome comparison using MinHash sketches. Developed to address the growing need for efficient analysis of large-scale genomic data, Sourmash offers a unique approach to sequence comparison that balances speed, accuracy, and computational efficiency. At its core, Sourmash employs a technique called FracMinHash, an extension of the MinHash algorithm, to create compact representations (sketches) of genomic sequences. These sketches allow for quick estimation of sequence similarity and containment, making Sourmash particularly useful for tasks such as Genome similarity searches, Metagenome analysis and decomposition, Taxonomic classification, Large-scale genomic database searches, Contamination detection in sequencing data.

Sourmash stands out from other genomic comparison tools due to its ability to handle data sets of vastly different sizes, making it especially valuable for metagenome analysis. It can efficiently compare a single genome against a complex metagenome as it uses scaled sketches instead of fixed length sketches or search for the presence of specific genomes within large metagenomic data sets. The tool is implemented as both a command-line application and a Python library, offering flexibility for various bioinformatics workflows. Sourmash is designed to be user-friendly while providing the power and scalability needed for modern genomic research.

The way sourmash works is straightforward. It breaks down DNA sequences into k-mers (subsequences of length k) as the fundamental unit of comparison. It creates compressed representations of sequences called ”signatures” by selecting a subset of k-mers based on their hash values [12].

2.4 BinDash 2.0

BinDash 2.0 is an advanced, multi-threaded software tool designed for ultra-fast and accurate genome search and comparisons at a massive scale. It's an updated version of the original BinDash, implementing several new MinHash algorithms and computational optimizations to enhance speed and efficiency while maintaining high accuracy. BinDash 2.0 is particularly useful for comparing large numbers of genomes, capable of performing approximately 0.1 trillion pairs of genome comparisons in about 1.8 hours on a decent computer cluster or several hours on personal laptops [14].

At its core, BinDash 2.0 utilizes a technique called b-bit one-permutation rolling MinHash with optimal/faster densification. This approach allows for efficient compression of genomic data into compact representations, similar to the sketches used in other tools like Sourmash. The software applies one-permutation MinHash to all k-mers in a genome, partitions the hash value universe into a predefined number of buckets, and extracts the smallest hash value in each bucket. It then uses densification algorithms to handle cases where buckets might be empty for some genomes but not others [14].

BinDash 2.0 stands out for its ability to balance speed and accuracy in large-scale genomic comparisons. It's particularly useful for tasks such as defining prokaryotic genome species boundaries and performing massive-scale genome similarity searches. The tool's efficiency makes it practical for researchers dealing with the ever-growing number of microbial genomes in public databases, enabling comparisons between millions or even billions of genomes. The implementation of advanced MinHash algorithms and computational optimizations, including the use of Simple Instruction Multiple Data (SIMD), positions BinDash 2.0 as a significant advancement in genomic comparison tools [14].

To illustrate the comprehensive approach of our study, we have developed a flow diagram

that outlines the key steps in our testing process. This diagram serves as a visual roadmap for our methodology, highlighting the systematic nature of our comparison of genome similarity tools.

A crucial step in the process is documenting the performance of each tool, noting key metrics such as processing time, comparison of the results from running the toools. This is followed by analyzing the recorded data using statistical methods to evaluate performance differences between tools, identifying strengths and weaknesses. Finally, based on this analysis, conclusions are drawn about the relative effectiveness of each tool, providing insights into their suitability for various genomic research applications.

Chapter 3

Objective

The objective of this study is to develop a Python-based pipeline for automated genome comparisons, facilitating systematic evaluations of genome similarity tools under consistent conditions. The pipeline executes FastANI, PyANI, Sourmash, and BinDash 2.0 in sequence, ensuring each tool processes identical data sets for unbiased comparisons. Key performance metrics, particularly execution time, are recorded to assess the computational efficiency of each tool.

In addition to time measurements, this study aims to evaluate and compare the similarity percentages produced by each tool. Initial analyses were conducted on a subset of five genomes to rank the results and identify optimal parameters, such as k-mer sizes for Sourmash and BinDash, to enhance their comparability with alignment-based tools like PyANI and FastANI. These findings provided a foundation for extending the study to a comprehensive dataset of 175 genomes.

The expanded analysis applies the optimal parameters identified in the smaller dataset to the larger dataset, enabling a thorough comparison of rank correlations between tools and further evaluations of computational efficiency, including average time per comparison. This approach ensures a holistic understanding of the performance and relative strengths of each tool, providing valuable insights into their applicability in genomic research and offering a framework for optimizing genome comparison methodologies.

Chapter 4

Materials

The data used in this study comprised 175 high-quality genome sequences sourced from the National Center for Biotechnology Information (NCBI) database. These genomes were carefully selected from two genome datasets provided by the Genome Taxonomy Database (GTDB): GTDB r207 (Revision 207) and GTDB r220 (Revision 220). Both datasets are renowned for their comprehensive curation of prokaryotic genomes, serving as a critical resource for genomic research and taxonomic studies.

The selection process began by pooling genomes from both GTDB revisions, totaling approximately 600,000 genomes across the two revisions. These genomes are curated to provide a single, high-quality set of genomes, reducing redundancy and ensuring accuracy in comparative analyses. From this initial pool, a stringent filtering step was applied based on the completeness percentage of each genome. Only genomes with a completeness score of 100%, as reported by the GTDB, were retained for this study. This filtering step ensured that the dataset consisted solely of high-quality genomes, eliminating incomplete or fragmented sequences that could compromise the reliability of the comparative analyses.

It is important to note that while the completeness scores were derived from GTDB's metadata, the genome sequences themselves were directly retrieved from NCBI. There were exactly 175 genomes in that revision whose completeness percentage was a 100%. The accession numbers for all 175 genomes are listed in the appendix for reference.

Chapter 5

Methods

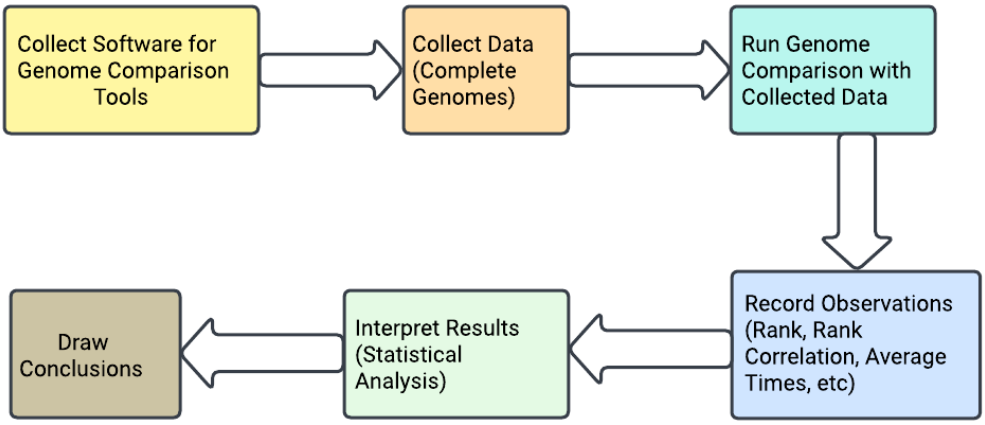

Figure 5.1: Six phases from collection of tools to drawing conclusions.

The diagram outlines a clear, step-by-step process for conducting the study. It begins with gathering the necessary software tools for genome comparison, ensuring they are the latest versions with all required dependencies. This is followed by acquiring a data set of complete genomes, focusing on high-quality, nearly complete sequences to ensure accurate comparisons. The third step involves using the collected genomic data to perform comparisons with each of the selected tools, adhering to standardized procedures for consistency.

The initial phase of our study involved the careful selection and acquisition of genome comparison tools. Based on an extensive literature review and consideration of current trends in bioinformatics, we chose four tools that represent a diverse range of approaches to genome comparison: PyANI, FastANI, Sourmash, and BinDash2.0. These tools were selected for

their varied methodologies, including both alignment-based and alignment-free techniques, to provide a comprehensive comparison of genome similarity assessment strategies.

PyANI, a Python module for calculating Average Nucleotide Identity (ANI), was chosen for its flexibility in offering multiple ANI calculation methods. FastANI was selected for its rapid, alignment-free approach to ANI calculation, utilizing MinHash sketching techniques for improved computational efficiency. Sourmash was included due to its use of MinHash sketches for genome and metagenome comparison, particularly its scalability with large data sets. Lastly, BinDash2.0 was chosen for its advanced binary sketching techniques in genomic distance estimation, offering potential improvements in both speed and accuracy.

Each tool was obtained from its official repository or website to ensure authenticity and up-to-date functionality. PyANI and Sourmash were installed using pip, the Python package installer, while FastANI and BinDash2.0 required compilation from source following the developers' instructions. All tools were installed on a Linux Ubuntu 20.04 workstation to maintain consistency in the testing environment.

In conducting genome comparisons using various software tools, there are two primary approaches that will be employed to ensure comprehensive analysis:

5.1 Database Creation and Querying

This approach involves creating a comprehensive database of carefully selected genomes using the capabilities provided by each tool. From a large repository of 172,000 genomes, we applied stringent filtering criteria to select 175 high-quality genomes. The only selection criterion was genome completeness, ensuring that our database consists of the most complete (100% in this case) genome samples available. This rigorous selection process allows us to

work with a refined set of genomes, optimizing the accuracy and reliability of our comparative analyses across different tools.

5.2 One-to-One Genome Comparison

The second approach focuses on direct one-to-one comparisons between pairs of genomes. This method involves selecting two genomes and running a comparison to obtain a straightforward result of their similarity. This approach is particularly useful for detailed analyses where specific genomic relationships are being investigated. It provides precise insights into the genetic relatedness of the selected genome pairs, allowing for targeted studies and validation of broader database-driven findings.

5.3 PyANI usage

PyANI is a Python package designed to calculate Average Nucleotide Identity (ANI) between whole-genome sequences. It provides several methods for ANI calculation, including ANIb (using BLAST), ANIm (using MUMmer), and others. PyANI is particularly useful for microbial genomics, allowing researchers to assess genomic similarity and delineate species boundaries. From this we can understand that pyANi provides four subcommands for genome comparison using ANIm, ANIb, aniblastall, and tetra [16].

The primary commands to run pyANI using each of the methods:

```
./average_nucleotide_identity.py -i tests/test_ani_data/ -o tests/test_ANIm_output -m AN
./average_nucleotide_identity.py -i tests/test_ani_data/ -o tests/test_ANIb_output -m AN
```

```
./average_nucleotide_identity.py -i tests/test_ani_data/ -o tests/test_ANIblastall_outpu
-m ANIblastall -g
./average_nucleotide_identity.py -i tests/test_ani_data/ -o tests/test_TETRA_output \
-m TETRA -g
```

In all of the four command above, the *-m* flag refers to one of the four ANI calculation methods that pyANI uses. The *-i* flag expects an input director as an argument that contains fasta files, and the *-o* flag indicates the output directory to which the results will be directed. When executing PyANI with the -i flag, the program identifies all genome files within the input directory and systematically compares each genome to every other genome. This results in a complete pairwise comparison, generating a square matrix of similarity results [16].

The matrix produced by PyANI provides a detailed overview of the genetic relatedness between all genomes in the data set. Each cell in the matrix represents the Average Nucleotide Identity (ANI) percentage between a pair of genomes, offering insights into their evolutionary relationships. This matrix format not only facilitates easy visualization of genomic similarities but also serves as a valuable resource for further analysis, such as clustering or phylogenetic studies. By leveraging PyANI's capabilities to perform exhaustive pairwise comparisons, researchers can gain a comprehensive understanding of the genomic landscape within their data set, enabling more informed conclusions about species boundaries and evolutionary dynamics [16].

5.4 FastANI usage

FastANI is a tool designed for rapid, alignment-free computation of whole-genome Average Nucleotide Identity (ANI). It is particularly efficient for processing large data sets due to its use of a MinHash-based approach, which significantly reduces computation time compared

to traditional ANI methods [15]. FastANI offers a straightforward and efficient approach to genome comparison, utilizing three primary types of commands to accommodate different analysis needs: **One-to-One Genome Comparison**: This command type is used for comparing a single query genome against a single reference genome. It is ideal for obtaining a direct measure of similarity between two specific genomes.

```
./fastANI -q [QUERY_GENOME] -r [REFERENCE_GENOME] -o [OUTPUT_FILE]
```

This command outputs the Average Nucleotide Identity (ANI) between the two specified genomes, providing a clear indication of their genetic relatedness [9].

One-to-Many Genome Comparison: This approach allows for the comparison of a single query genome against multiple reference genomes listed in a file. It is useful for assessing the similarity of one genome across a range of references. The command syntax is:

```
./fastANI -q [QUERY_GENOME] --rl [REFERENCE_LIST] -o [OUTPUT_FILE]
```

The reference list file contains paths to multiple reference genomes, enabling comprehensive analysis with a single command execution.

Many-to-Many Genome Comparison: For large-scale analyses, FastANI can compare multiple query genomes against multiple reference genomes, each specified in separate list files. This method is efficient for generating a complete similarity matrix across large data sets. The command syntax is:

```
./fastANI --ql [QUERY_LIST] --rl [REFERENCE_LIST] -o [OUTPUT_FILE]
```

By processing lists of genomes, this command facilitates high-throughput genomic comparisons, essential for extensive genomic studies. These command types enable flexible and

scalable genome comparison analyses, making FastANI a powerful tool for genomic research. By leveraging these commands, researchers can efficiently assess genetic similarities across a wide array of genomic data, supporting various applications from species delimitation to evolutionary studies [9].

5.5 Sourmash usage

Sourmash is a bioinformatics tool designed to quickly estimate genome similarity using MinHash sketches. It is particularly useful for comparing large genomic data sets efficiently, as it reduces the computational load by summarizing sequence data into compact sketches. To effectively use Sourmash for genome comparison, the first step is to convert FASTA files into signatures. This process involves creating MinHash sketches, which are compact representations of the sequences that facilitate quick similarity comparisons [17]. There are two primary methods for generating these signatures:

5.5.1 Converting FASTA Files into Signatures

Single FASTA File to Signature: To convert a single FASTA file into a signature, you can use the sourmash compute command. This command generates a MinHash sketch for the specified genome, which can then be used for comparison. Example command:

```
sourmash compute -k 31 --scaled 1000 -o genome.sig genome.fna
```

In this command, *-k* specifies the k-mer size, –scaled sets the scaling factor, *-o* defines the output file for the signature, and genome.fna is the input FASTA file. **Multiple FASTA Files to Signature Database**: To convert multiple genome files into a database of sig-

natures, you can use the same sourmash compute command but apply it to a directory containing multiple FASTA files. Example command:

```
sourmash compute -k 31 --scaled 1000 -o all_genomes.sig /path/to/genomes/*.fna
```

This command processes all FASTA files in the specified directory, creating a single output file $all_genomes.sig$ that contains the signatures for all genomes.

By using these commands, we can prepare genomic data for efficient comparison using Sourmash. The generated signatures allow for rapid similarity assessments, facilitating large-scale genomic analyses. Once the signatures are created, we can proceed with comparing them to estimate genome similarity, leveraging Sourmash's capabilities for fast and scalable genomic comparisons[12].

5.5.2 Performing Genome Comparison with Sourmash

After the conversion of FASTA files into signatures, the subsequent step involves performing genome comparisons. Sourmash utilizes MinHash sketches to efficiently estimate genome similarity, which have been obtained from the previous step. The sourmash compute function can be employed in three primary ways:

Single Signature Comparison: A single genome signature can be compared against another using the sourmash compare command. This command calculates the similarity between the specified signatures and outputs a similarity matrix. Example command:

```
sourmash compare --csv similarity_matrix.csv genome1.sig genome2.sig
```

This command generates a CSV file $similarity_matrix.csv$ containing the similarity scores between the specified genome signatures.

Multiple Signature Comparison: For the comparison of multiple genome signatures, a list of signature files can be specified. Sourmash computes pairwise similarities for all specified signatures. Example command:

```
sourmash compare --csv similarity_matrix.csv genome1.sig genome2.sig genome3.sig
```

This command creates a similarity matrix for all the provided signatures, allowing for the assessment of genomic relationships across the data set.

Using a Signature Database: When a database of signatures has been created, comparisons against this database can be performed to quickly assess similarities across a large number of genomes.

```
sourmash compare --csv similarity_matrix.csv /path/to/signature_database/*.sig
```

This approach is efficient for large-scale analyses, as the precomputed signatures are leveraged to facilitate rapid comparisons. By employing these commands, comprehensive genome comparisons with Sourmash can be performed, utilizing its speed and efficiency to analyze large genomic data sets. The resulting similarity matrices provide valuable insights into genetic relationships, supporting further analysis and interpretation in research[17].

5.6 BinDash 2.0 usage

BinDash 2.0 is a tool designed for efficient genome comparison using advanced sketching techniques. It leverages binary sketches to estimate genomic distances quickly, making it suitable for large-scale analyses. Running BinDash 2.0 for genome comparison is a straightforward process that is divided into two main steps.

5.6.1 Converting to Sketches

The first step involves converting a single genome or a group of genomes into binary sketches. This conversion is essential as it creates compact representations of the genomic data, which are used for efficient comparison. There are two primary methods for creating sketches: **Single Genome to Sketch**: A single genome can be converted into a binary sketch using the following command:

```
bindash sketch -i genome.fna -o genome.sketch
```

In this command, *-i* specifies the input genome file, and *-o* defines the output file for the binary sketch. **A list of genomes to sketches** A list of genomes whose names are stored in a text file can be converted into sketched corresponding to the names in the text file.

```
bindash sketch --listfname=name.txt --outfname=genome_sketch
```

This method is particularly useful for handling multiple genomes simultaneously, creating a comprehensive set of sketches that can be used for subsequent genomic distance calculations. By organizing genome paths in a list file, the process becomes streamlined and efficient, facilitating large-scale genomic analyses [14].

5.6.2 Computing genomic similarity using the sketches

Once the sketches are created, BinDash 2.0 computes the genomic distances between them. This step involves comparing the binary sketches to estimate the genetic distances, providing insights into the genomic relationships between the samples. **Comparison between two sketches**: There is a simple command to compute the distance between two sketches which correspond to two genomes.

```
bindash dist genomeA.sketch genomeB.sketch
```

This command simply computes the genomic distance between two binary sketches, genomeA.sketch and genomeB.sketch.

Computing genomic distances between muliple sketches: The simple command to calculate the genomic distances:

```
bindash dist --outfname=dist.txt genome_query_sketch genome_db_sketch
```

Here the distances between the sketches in $genome_query_sketch$ and $genome_db_sketch$ are outputed to *dist.txt* providing a measure of similarity between the query genomes and the database genomes. By following this sequence, BinDash 2.0 efficiently processes multiple genomes to generate sketches and then computes the distances between them, facilitating large-scale genomic comparisons [14].

Following installation of these tools, we conducted thorough verification tests to ensure each tool was correctly installed and functioning as expected. This involved running tests using sample data sets provided by the developers or small, well-characterized genomic sequences. These verification steps were crucial in confirming that each tool was producing expected outputs before proceeding with our full-scale analysis, thus establishing a solid foundation for our comparative study of genome similarity tools [10][18].

5.7 Program for Automated Genome Comparison

To conduct a robust and consistent comparative analysis across multiple genome similarity tools, we began developing a Python program designed to automate the entire genome comparison process for four selected tools: FastANI, PyANI, Sourmash, and BinDash 2.0.

The initial version of the program uses a smaller subset of the original dataset, five random genomes chosen from the set of 175 fully sequenced genomes. This subset provided a preliminary view of the comparative results and allowed us to understand how each tool measures genome similarity before scaling the analysis.

The primary purpose of this automation is to standardize the comparison process across the selected tools, ensuring that each genome pair is processed under identical conditions. This approach provides uniform and reproducible results while significantly reducing manual workload and minimizing potential human error. The program not only facilitates rapid comparisons but also prepares us to scale to the full dataset of 175 genomes.

The current workflow of the program consists of the following steps:

- **Data Input**: The program accepts a list of genomes in FASTA format. For initial testing and parameter tuning, we selected a subset of ten genomes. This reduced dataset enabled iterative testing and parameter adjustments with lower computational overhead.
- **Integration of Tools and Parameter Settings**: Each of the four tools—FastANI, PyANI, Sourmash, and BinDash—has unique input requirements and parameters, so the program is designed to configure and execute each tool accordingly. The program can adjust key parameters for k-mer-based tools, such as k-mer size (k) and sketch density (scaled), allowing for flexible optimization and eventual scaling.
- **Pairwise Genome Comparisons**: The program generates all unique pairs of the selected genomes and runs each tool in sequence for each pair. The results are collected and stored in a structured format, with rows representing each genome pair and columns for each of the four tools.
- **Result Aggregation and Table Formation**: The program compiles the similar-

ity results for each tool into a consolidated DataFrame, making it easy to interpret and compare results across methods. This output provides a comprehensive view of pairwise similarity scores and enables direct comparison between alignment-based and k-mer-based methods.

- **Initial Observations and Future Expansion**: By running this process on a subset of five genomes, we were able to observe initial patterns in similarity scores, which informed parameter tuning. This phase allows us to identify potential discrepancies or correlations between the methods, preparing us for full dataset processing.

Chapter 6

Results

To assess genome similarity across four different computational tools—PyANI, FastANI, Sourmash, and BinDash—we began by conducting pairwise comparisons on a representative subset of five genomes. This subset provided a practical starting point to observe baseline similarity values across the methods and allowed us to identify any initial discrepancies or trends prior to expanding to a larger dataset.

For this initial comparison, each tool was executed using its default parameters to establish a baseline for similarity values. Table 1 summarizes these initial pairwise similarity scores, with results expressed as similarity percentages for PyANI and FastANI, and as Jaccard similarity scores for Sourmash and BinDash. As expected, PyANI and FastANI yielded high similarity values, often between 85% and 99% for closely related genomes, reflecting their sensitivity to nucleotide-level similarities across homologous regions.

However, the initial results for Sourmash and BinDash displayed significantly lower similarity scores for the same genome pairs. This discrepancy was most pronounced for genome pairs that PyANI and FastANI identified as highly similar. As shown in Table 6.1, PyANI and FastANI consistently yield high similarity percentages for closely related genomes, with values ranging from approximately 82% to nearly 100%, depending on the genome pair. These alignment-based methods provide a direct measure of nucleotide similarity, making them highly accurate for closely related genomes.

In contrast, Sourmash and BinDash, which use k-mer-based, alignment-free methods, display considerably lower similarity scores under default parameters. Sourmash's Jaccard similarity scores, using the default k-mer size of k=21, range from as low as 0.012 to a maximum of 0.823. Similarly, BinDash's scores, also presented as Jaccard similarity, are notably low, with most values below 0.05 except for the highest-scoring pairs, such as 2v3 and 3v4. These discrepancies highlight a fundamental difference in how k-mer-based methods capture genome similarity compared to alignment-based ANI calculations.

Genome Pair	**PyANI %**	**FastANI %**	**Sourmash ($k = 21$)**	**BinDash (Default k)**
Genome 1v2	84.869 %	83.204 %	0.019	84/2048 (0.041)
Genome 1v3	84.821 %	83.221 %	0.018	90/2048 (0.044)
Genome 1v4	84.828 %	83.324 %	0.018	72/2048 (0.035)
Genome 1v5	83.797 %	81.155 %	0.012	66/2048 (0.032)
Genome 2v3	99.943 %	99.887 %	0.823	1736/2048 (0.848)
Genome 2v4	95.907 %	95.811 %	0.252	632/2048 (0.309)
Genome 2v5	84.205 %	82.031 %	0.014	69/2048 (0.034)
Genome 3v4	95.851 %	95.675 %	0.258	634/2048 (0.310)
Genome 3v5	84.073 %	81.899 %	0.013	71/2048 (0.035)
Genome 4v5	84.179 %	81.945 %	0.014	76/2048 (0.037)

Table 6.1: Pairwise genome comparison values across the four tools

The low Jaccard similarity scores in Sourmash and BinDash indicate that the default k-mer size (e.g., k=21 for Sourmash) may not be optimal for accurately reflecting genome similarity in closely related genomes. In alignment-based methods like PyANI and FastANI, high similarity percentages are expected for homologous regions. However, k-mer-based methods may require smaller k-mer sizes to capture sufficient shared sequences between similar genomes, especially when comparing highly similar or closely related genomes.

Therefore, to improve the comparability of Sourmash and BinDash results with PyANI and FastANI, we explored smaller k-mer values for both k-mer-based tools. Adjusting the k-mer size in Sourmash and BinDash has the potential to increase Jaccard similarity scores, bringing them closer to the ANI percentages reported by alignment-based tools. The following sections present the results of this parameter tuning process and the resulting optimized Jaccard similarity scores.

As shown in Table 6.2, Jaccard similarity scores in Sourmash increased consistently as the k-mer size decreased. Starting with k=15, the Jaccard values are relatively low across all genome pairs. As the k-mer size is reduced to 13 and then to 11, there is a notable rise in similarity values, indicating that smaller k-mer sizes capture more shared k-mers between similar genomes, which aligns with expectations.

k-value	**1v2**	**1v3**	**1v4**	**1v5**	**2v3**	**2v4**	**2v5**	**3v4**	**3v5**	**4v5**
15	0.065	0.065	0.067	0.049	0.840	0.344	0.058	0.349	0.059	0.062
13	0.187	0.190	0.198	0.173	0.855	0.439	0.176	0.441	0.178	0.194
11	0.575	0.586	0.595	0.599	0.929	0.713	0.580	0.729	0.598	0.628
10	0.846	0.833	0.865	0.871	0.964	0.881	0.844	0.872	0.846	0.883
9	0.973	0.982	0.991	0.973	0.991	0.964	0.982	0.973	0.973	0.982

Table 6.2: Sourmash Jaccard Similarity Scores for Varying k-mer Sizes

For k=10, the similarity values are significantly higher, and in many cases, approach the percentage similarity scores provided by PyANI and FastANI. This result suggests that k=10 may be an optimal k-mer size for Sourmash when aiming to produce similarity values comparable to alignment-based methods like PyANI and FastANI. Specifically, for genome pairs such as 2v3 and 3v4, which showed high similarity in ANI scores, the Jaccard values for k=10 are notably close to those percentage values, reinforcing the appropriateness of this k-mer size.

At k=9, the similarity scores are very close to 1.0 for most pairs, indicating nearly complete overlap in shared k-mers. This dramatic increase suggests that k=9 may capture too many shared k-mers, potentially leading to an overestimation of similarity for closely related genomes and diminishing the discriminatory power for detecting slight differences. Thus, while k=9 maximizes similarity scores, k=10 strikes a balance between sensitivity and comparability with ANI-based methods.

These findings underscore the importance of k-mer size selection in k-mer-based methods like Sourmash. By adjusting k-mer size, we can achieve similarity estimates that align more closely with alignment-based approaches, making k=10 a reasonable choice for achieving comparability with ANI-based tools in genome similarity studies.

k-value	1v2	1v3	1v4	1v5	2v3	2v4	2v5	3v4	3v5	4v5
15	0.064	0.067	0.066	0.046	0.849	0.333	0.054	0.336	0.054	0.056
13	0.174	0.178	0.186	0.170	0.857	0.427	0.168	0.435	0.172	0.187
11	0.565	0.568	0.591	0.590	0.911	0.697	0.587	0.702	0.598	0.627
10	0.854	0.857	0.868	0.875	0.957	0.884	0.853	0.893	0.867	0.883
9	0.975	0.978	0.979	0.978	0.989	0.985	0.983	0.987	0.985	0.987

Table 6.3: BinDash Jaccard Similarity Scores for Varying k-mer Sizes

The pairwise comparison results for BinDash across different k-mer sizes, presented in Table 6.3, reveal a clear trend: Jaccard similarity scores increase as the k-mer size decreases. At larger k-mer sizes, such as k=15 and k=13, BinDash similarity scores remain relatively low, especially for more distantly related genome pairs. This trend of increasing similarity continues as we reduce the k-mer size, reaching higher values around 0.4–0.6 at k=11.

With k=10, a marked improvement in similarity scores is observed, with values that closely approximate the ANI percentages provided by alignment-based tools like PyANI and FastANI. For example, the similarity score for genome pair 2v3 reaches 0.957, closely aligning with the high ANI values reported by PyANI and FastANI. This makes k=10 a favorable choice for BinDash, providing similarity estimates that better match those of alignment-based tools while still benefiting from the efficiency of a k-mer-based approach.

A similar trend was observed in Sourmash, where k=10 also emerged as the optimal k-mer size, significantly enhancing Jaccard similarity scores to levels comparable with ANI percentages. At k=9, both BinDash and Sourmash produced similarity values close to 1.0 for most genome pairs, indicating near-complete overlap in shared k-mers. This suggests that k=9 may overestimate similarity for closely related genomes, reducing the ability to discern subtle differences.

The consistency in results between BinDash and Sourmash demonstrates the importance of tuning k-mer size in k-mer-based, alignment-free genome similarity tools. For both tools, k=10 provides an optimal balance, yielding similarity scores that closely match those of alignment-based methods like PyANI and FastANI. This k-mer size captures sufficient shared sequence information to reflect true genomic similarity, while avoiding the oversensitivity observed at k=9. Thus, by selecting k=10, both BinDash and Sourmash can serve as efficient alternatives for genome similarity estimation, achieving comparability with traditional ANI-based methods.

Table 6.4 presents the pairwise genome similarity scores across all four tools, with the same PyANI and FastANI values from the initial comparison. However, for Sourmash and BinDash, the table now includes optimized similarity scores using k=10, which was identified as the most effective k-mer size to achieve comparability with ANI-based methods. This optimized configuration brings Sourmash and BinDash values closer to the percentages reported by PyANI and FastANI, demonstrating that tuning the k-mer size can significantly improve the alignment-free methods' ability to approximate alignment-based similarity scores. In order to provide a clear understanding of the relative similarity scores across the four tools—PyANI, FastANI, Sourmash, and BinDash—a ranking system was applied to the raw comparison values. Although each tool uses distinct methodologies (alignment-based or k-mer-based) and therefore produces unique absolute similarity values for a given genome pair, the relative rankings can help reveal patterns and relationships across tools.

Genome Pair	**PyANI (%)**	**FastANI (%)**	**Sourmash (k = 10)**	**BinDash (k = 10)**
Genome 1v2	84.869%	83.204%	0.846	0.854
Genome 1v3	84.821%	83.221%	0.833	0.857
Genome 1v4	84.828%	83.324%	0.865	0.868
Genome 1v5	83.797%	81.155%	0.871	0.875
Genome 2v3	99.943%	99.887%	0.964	0.957
Genome 2v4	95.907%	95.811%	0.881	0.884
Genome 2v5	84.205%	82.031%	0.844	0.853
Genome 3v4	95.851%	95.675%	0.872	0.893
Genome 3v5	84.073%	81.899%	0.846	0.867
Genome 4v5	84.179%	81.945%	0.883	0.883

Table 6.4: Pairwise Genome Similarity Scores Across Tools with Optimized k-mer Size (k=10) for Sourmash and BinDash

For each tool, the similarity values for all genome pairs were ordered from highest to lowest, with the highest similarity value assigned a rank of 1 and the lowest a rank of 10. This ranking is displayed in Table 6.5, which illustrates the ”relative order” or ”relative ranking” of similarity scores within each tool. By analyzing these ranks instead of the raw values, we can observe whether certain genome pairs consistently receive high or low similarity scores across different tools, despite variations in absolute values. This comparative approach provides a more unified perspective on how each tool perceives genome similarity and helps in identifying consistent patterns or anomalies across methodologies

Genome Pair	**PyANI**	**FastANI**	**Sourmash (k=10)**	**BinDash (k=10)**
Genome 1v2	6	6	7	6
Genome 1v3	7	5	9	5
Genome 1v4	5	4	5	4
Genome 1v5	10	10	4	3
Genome 2v3	1	1	1	1
Genome 2v4	2	2	3	2
Genome 2v5	9	9	8	8
Genome 3v4	3	3	2	2
Genome 3v5	8	8	7	7
Genome 4v5	4	7	3	3

Table 6.5: Relative Rankings of Similarity Scores for Each Tool

6.1 Rank Correlation Between Tools for Relative Similarity Assessment

To better understand the comparative performance of different genome similarity tools, it is essential to examine the rank correlation between each pair of tools. By focusing on rank correlation, we can evaluate the relative order of similarity scores that each tool assigns to genome pairs, rather than absolute similarity values. This approach offers a way to understand whether two tools generally agree on which genome pairs are most similar, even if they differ in their specific values.

Establishing rank correlation provides valuable insights into how consistent or divergent the tools are when measuring genome similarity. A high or a positive rank correlation would indicate that the tools generally assign similar rankings to genome pairs, suggesting they have similar perspectives on relative genome similarity. In contrast, a low or a negative rank correlation would highlight discrepancies, suggesting that the tools may have fundamentally different interpretations of similarity based on their underlying methodologies (e.g., alignment-based vs. k-mer-based).

6.1.1 Kendall's Tau Coefficient for Pairwise Tool Comparisons

To quantify the rank correlation between tools, we use Kendall's τ coefficient, a statistical measure specifically designed for ordinal association. Kendall's τ is ideal for evaluating the consistency of rankings, as it assesses how well the order of similarity scores aligns across tools. A positive Kendall's Tau value indicates agreement between rankings, while a negative value would suggest an inverse relationship. Table 6.6 provides an overview of the Kendall's Tau values calculated for pairwise comparisons across the four genome similarity tools.

6.1.2 Spearman's Rank Correlation for Pairwise Tool Comparisons

To further examine the alignment in relative rankings between tools, we also calculated Spearman's rank correlation coefficient (ρ) for each pair of tools. Like Kendall's Tau, Spearman's ρ measures the degree to which the ranks are consistent between tools. However, Spearman's ρ captures linear monotonic relationships, offering additional insight into how similarly the tools rank genome pairs. Table 6.7 summarizes the Spearman's rho values for pairwise comparisons among the four genome similarity tools.

Tool Pair	**Kendall's τ**	**Spearman's ρ**
PyANI vs. FastANI	0.9111	0.9636
PyANI vs. Sourmash	0.3333	0.4303
PyANI vs. BinDash	0.3333	0.4788
FastANI vs. Sourmash	0.3333	0.4061
FastANI vs. BinDash	0.4222	0.5273
Sourmash vs. BinDash	0.7333	0.8909

Table 6.6: Kendall's Tau values for Pairwise Tool Comparisons

Tool Pair	**Spearman's ρ**
PyANI vs. FastANI	0.9636
PyANI vs. Sourmash	0.4303
PyANI vs. BinDash	0.4788
FastANI vs. Sourmash	0.4061
FastANI vs. BinDash	0.5273
Sourmash vs. BinDash	0.8909

Table 6.7: Spearman's rho values for Pairwise Tool Comparisons

Based on the Kendall's Tau and Spearman's rho values presented in the tables, we can conclude that the rank correlation between FastANI and PyANI is notably high, with values exceeding 0.9 in both metrics. This strong correlation is expected, as both FastANI and PyANI utilize alignment-based methods, which often yield consistent similarity rankings. Similarly, Sourmash and BinDash exhibit a high rank correlation (above 0.7), reflecting the consistency between these k-mer-based tools in their approach to genome similarity.

The rank correlation between pairs consisting of one alignment-based tool (FastANI or PyANI) and one k-mer-based tool (Sourmash or BinDash) is lower, as anticipated, yet remains positive. This indicates that, despite the methodological differences, there is some alignment in how these tools rank genome pairs, though not to the same extent as within-tool type comparisons. Overall, these results demonstrate that tools within the same methodological class (alignment-based or k-mer-based) show a stronger rank correlation, while cross-class comparisons reflect a moderate degree of rank correlation, capturing relative similarities to some extent.

6.2 Results for Full Dataset of 175 Genomes

Building upon the baseline analysis conducted on a smaller subset of five genomes, we extended the genome similarity comparisons to a larger dataset of 175 genomes. This expansion resulted in $\binom{175}{2}$ which is the equivalent of 15225 unique pairwise comparisons. Given the scale of this analysis, it was no longer practical to present individual similarity scores or their corresponding ranks in a tabular format, as was done for the smaller dataset.

However, analyzing these pairwise comparisons revealed trends across the four tools which were similar to the observations made in the smaller subset. As observed in the smaller subset, alignment-based methods (PyANI and FastANI) consistently reported higher similarity

percentages, ranging from approximately 85% to nearly 100% for closely related genomes. In contrast, k-mer-based methods (Sourmash and BinDash) displayed lower Jaccard similarity scores under default parameters, highlighting the need for parameter optimization. Drawing from our earlier analysis of a smaller subset of genomes, we observed that reducing the k-mer size to $k = 10$ significantly improved the similarity scores for both Sourmash and BinDash. Specifically, this optimization brought their similarity values closer to those reported by alignment-based methods such as PyANI and FastANI, effectively narrowing the gap between these two methodological approaches.

Based on these findings, we applied the optimized $k = 10$ parameter to Sourmash and BinDash for the larger dataset of 175 genomes. The results demonstrated a marked improvement in similarity scores for both tools, achieving values that were not only higher but also more comparable to the percentages reported by PyANI and FastANI. This adjustment reinforces the importance of parameter tuning in k-mer-based methods to achieve alignment with traditional ANI-based tools.

Tool Pair	**Kendall's Tau**	**Spearman's rho**
PyANI vs. FastANI	0.8625	0.9630
PyANI vs. Sourmash	0.3507	0.4942
PyANI vs. BinDash	0.4721	0.6327
FastANI vs. Sourmash	0.3276	0.4679
FastANI vs. BinDash	0.4375	0.6011
Sourmash vs. BinDash	0.5290	0.6967

Table 6.8: Kendall's Tau values for Pairwise Tool Comparisons on 175 Genomes

6.2.1 Rank Correlation Analysis for 175 Genomes

One of the primary objectives of this study was to evaluate the consistency in how different genome similarity tools rank genome pairs based on their similarity scores. This was particularly critical for the expanded dataset, where we sought to determine whether the patterns observed in the smaller subset would hold across a much larger and more diverse set of genomes. To achieve this, we calculated Kendall's Tau and Spearman's rho coefficients for the rankings produced by each pair of tools.

As presented in Table 6.8, Kendall's Tau coefficients indicate a strong rank correlation between alignment-based tools (pyANI and FastANI), with a value of 0.8625. This high correlation aligns with their shared methodological foundation. Similarly, the k-mer-based tools (Sourmash and BinDash) exhibit moderate correlation (0.5290), reflecting their comparable approaches to genome similarity estimation. In contrast, rank correlations between alignment-based tools and k-mer-based tools (e.g., pyANI vs Sourmash) are lower but remain positive. This suggests some alignment in rankings, albeit with methodological differences influencing the results.

Tool Pair	**Spearman's rho**
pyANI vs FastANI	0.9630
pyANI vs Sourmash	0.4942
pyANI vs BinDash	0.6327
FastANI vs Sourmash	0.4679
FastANI vs BinDash	0.6011
Sourmash vs BinDash	0.6967

Table 6.9: Spearman's rho values for Pairwise Tool Comparisons

Table 6.9 provides Spearman's rho values, which capture linear monotonic relationships in the rankings. Similar to Kendall's Tau, Spearman's rho indicates strong agreement between pyANI and FastANI (0.9630), and moderate agreement between Sourmash and BinDash (0.6967). However, the cross-class comparisons again reveal lower correlations, such as 0.4942 for pyANI vs Sourmash.

These observations are consistent with the trends identified in the smaller subset analysis, where alignment-based tools demonstrated higher rank correlations, and k-mer-based tools showed moderate alignment. The rank correlations for the larger dataset further validate these patterns. Overall, Kendall's Tau and Spearman's rho analyses highlight strong rank correlations within tool classes (alignment-based vs k-mer-based) and moderate correlations across classes. These findings underscore the methodological differences between alignment-based and k-mer-based approaches while revealing some level of consistency in ranking genome pairs.

6.2.2 Time Analysis of Genome Comparison Tools

One of the critical aspects of this study is the computational feasibility of the genome similarity tools when applied to large datasets. As the dataset size increases, the computational time required for pairwise comparisons grows significantly. Since the comparisons involve all possible pairs of genomes, the number of comparisons increases quadratically with the dataset size. For instance, moving from a smaller subset of 5 genomes ($\binom{5}{2} = 10$ comparisons) to the full dataset of 175 genomes ($\binom{175}{2} = 15,225$ comparisons) results in an enormous increase in computational demand.

This exponential growth in time required for pairwise comparisons poses a major challenge, particularly for tools that rely on computationally intensive alignment-based methods. Run-

ning the program for all four tools on the full dataset was a monumental effort, taking a significant amount of time, particularly for alignment-based approaches. Recording and analyzing the time taken by each tool thus became an integral part of the study, offering valuable insights into their scalability and computational efficiency.

The subsequent analysis highlights the total time taken by each tool to complete the 15,225 pairwise comparisons, along with the average time required for a single comparison. These metrics provide a clear perspective on the computational trade-offs between alignment-based methods (pyANI and FastANI) and k-mer-based methods (Sourmash and BinDash), as well as the feasibility of using these tools for large-scale genomic studies.

The total time taken for genome comparisons across the four tools, as summarized in Table 6.10 , highlights the significant differences in computational requirements between alignment-based and k-mer-based methods. Alignment-based tools like pyANI and FastANI, which involve detailed sequence alignment processes, naturally require more time. This is evident in the total time taken, with pyANI taking approximately 8.39 hours and FastANI around 3.43 hoursSow for the dataset.

Tool	**Total Time (seconds)**	**Avg. Time per Comparison (seconds)**
pyANI	30217.14 (8.39 hours)	1.98
FastANI	12339.25 (3.427 hours)	0.81
Sourmash	32.18	0.0021 (2.1 ms)
BinDash	3.80	0.00025 (0.25 ms)

Table 6.10: Total time taken by each tool for 15,225 pairwise genome comparisons.

In contrast, k-mer-based tools like Sourmash and BinDash are designed for efficiency, utilizing alignment-free techniques that are computationally less intensive. This efficiency is reflected in their remarkably shorter runtimes, with Sourmash completing the task in just 32.18 seconds and BinDash in a mere 3.80 seconds. These results underscore the trade-offs between computational speed and methodological complexity, with alignment-based tools offering high precision at the cost of longer runtimes, while k-mer-based tools prioritize rapid computation.

To compare the computational efficiency of the four tools fairly, we calculated the average time taken per genome comparison as shown in Table 6.11. For consistency, we normalized the total runtime for all tools by dividing by the number of unique pairwise genome comparisons (($\binom{175}{2} = 15,225$ comparisons). This approach assumes uniform processing for all comparisons.

The average time per genome comparison highlights a significant contrast between alignment-based and k-mer-based methods. For PyANI and FastANI, the average times per comparison are 1.98 seconds and 0.81 seconds, respectively. These relatively higher times are reflective of the computational complexity involved in alignment-based methods, which require extensive sequence alignment and comparison across homologous regions. This complexity increases significantly as the dataset size grows, making these methods computationally intensive for large-scale analyses.

Tool	**Avg. Time per Comparison (seconds)**
pyANI	1.98
FastANI	0.81
Sourmash	0.0021 (2.1 ms)
BinDash	0.00025 (0.25 ms)

Table 6.11: Average time taken per genome comparison for each tool. Values are computed by dividing the total runtime for each tool by the number of unique pairwise comparisons $(15,225)$.

In contrast, k-mer-based methods like Sourmash and BinDash demonstrate remarkable efficiency, with average times of 0.0021 seconds (2.1 milliseconds) and 0.00025 seconds (0.25 milliseconds) per comparison, respectively. These tools leverage alignment-free approaches that involve computing Jaccard similarity scores based on shared k-mers, avoiding the need for computationally expensive alignment operations. This efficiency makes k-mer-based methods particularly well-suited for rapid preliminary analyses or large-scale comparisons where computational resources may be a limiting factor.

The disparity in average times also highlights the trade-offs inherent to these methodologies. While k-mer-based tools excel in speed, their reliance on approximations may impact their sensitivity and accuracy, especially for highly similar genomes. On the other hand, the precision of alignment-based methods comes at the cost of significantly higher computational requirements, underscoring their suitability for analyses where accuracy is paramount.

By presenting the average times, this study underscores the importance of tool selection based on the specific goals of the genome comparison task, balancing the trade-offs between computational efficiency and methodological precision.

Initially, for the default k-mer values, the similarity scores generated by Sourmash and BinDash were significantly lower. However, this does not indicate an issue of accuracy but instead reflects the fundamental differences in how alignment-based and alignment-free tools operate. Alignment-based tools, such as pyANI and FastANI, work by breaking the genome sequences into fragments and matching these fragments to the closest regions of the reference genome. Specifically, pyANIb employs the BLAST algorithm, while FastANI uses a hashing algorithm to identify orthologous matches. The similarity percentages are then averaged across all matching regions, resulting in the ANI percentage. Since only matching pairs contribute to the calculation, the resulting ANI values tend to be on the higher side for closely related genomes.

In contrast, alignment-free tools like Sourmash and BinDash employ a different approach. After splitting the nucleotide sequences into non-overlapping k-mers of length k, both tools select a representative subset of k-mers. The Jaccard index is then calculated by dividing the size of the common k-mers (intersection) between the two representative subsets by the size of all k-mers in the union of both subsets. Because the denominator includes all representative k-mers from both genomes, it is often much larger than the numerator (the intersection), leading to significantly lower Jaccard values compared to ANI percentages.

To clarify, alignment-based tools such as pyANI and FastANI focus solely on matching pairs to compute ANI percentages, resulting in higher similarity values. In contrast, Sourmash and BinDash consider all representative k-mers, leading to substantially lower Jaccard values. In this research, the k-mer size for Sourmash and BinDash was adjusted to k=10 to observe how Jaccard values fluctuate in the higher ranges. However, this adjustment was not intended to directly correlate Jaccard values with ANI percentages, as they are fundamentally different measures. The goal was to explore the behavior of Jaccard values under specific parameter settings while maintaining methodological rigor.

Chapter 7

Conclusion

This study provides a comprehensive evaluation of genome comparison tools, addressing their performance in terms of accuracy, computational efficiency, and methodological differences. By analyzing both alignment-based methods (pyANI and FastANI) and k-mer-based methods (Sourmash and BinDash 2.0), this thesis highlights key insights into their strengths, limitations, and applications in genomic research.

The results indicate that alignment-based tools, pyANI and FastANI, consistently produce high similarity percentages for closely related genomes, demonstrating their precision in capturing nucleotide-level similarities. However, these tools also require significantly higher computational resources, with pyANI taking approximately 8.39 hours and FastANI taking 3.43 hours to process 175 genomes. These findings emphasize their suitability for analyses where accuracy is prioritized over computational efficiency.

In contrast, k-mer-based tools, Sourmash and BinDash, provide a much faster alternative, completing genome comparisons in mere seconds. While their default parameters resulted in lower similarity values, parameter optimization—specifically selecting a k-mer size of 10—significantly improved their performance, bringing their similarity scores closer to those of alignment-based tools. This optimization underscores the importance of parameter tuning in k-mer-based methods to achieve meaningful results.

Rank correlation analyses further revealed strong concordance between tools within the

same methodological category. Kendall's Tau and Spearman's rho coefficients for alignment-based tools exceeded 0.85, indicating highly consistent rankings. Similarly, the k-mer-based tools displayed moderate rank correlations, reflecting their shared approach. Cross-category correlations were lower but still positive, demonstrating some alignment in how these tools rank genome pairs despite their fundamental differences.

A notable example of combining the strengths of alignment-free and alignment-based tools is the LINflow computational pipeline, as described in "LINflow: a computational pipeline that combines an alignment-free with an alignment-based method to accelerate generation of similarity matrices for prokaryotic genomes"[1]. This tool leverages the speed of an alignment-free method (Sourmash) and the precision of an alignment-based method (PyANI) to efficiently compute genomic similarities. LINflow sequentially adds genomes to a dataset, assigns Life Identification Numbers (LINs) to each genome based on its similarity to others, and uses these LINs to infer ANI percentages rather than calculating and storing them directly.

In their study, the researchers validated the effectiveness of LINflow by using Pearson correlation coefficients to compare its performance with Sourmash, PyANI, and FastANI. Their use of statistical correlation measures, such as Pearson coefficients, served as a direct inspiration for employing tools like Kendall's Tau and Spearman's Rho in this thesis. This paper highlights how alignment-free and alignment-based tools can complement each other to achieve both speed and precision, a concept that aligns with the objectives of this thesis.

In this thesis, I have compared four genome comparison tools: PyANI, FastANI, Sourmash, and BinDash. PyANI and FastANI are alignment-based tools that calculate ANI percentage values, while Sourmash and BinDash provide Jaccard indices. Although ANI percentages are fundamentally different from Jaccard indices, this thesis optimized the k-mer values for Sourmash and BinDash to make their Jaccard values relatively comparable to ANI percentages.

A promising direction for future work is to explore the conversion of Jaccard indices from Sourmash and BinDash into percentage values of ANI. The latest version of Sourmash already offers the ability to convert its Jaccard values into ANI percentages, which could be a significant step forward. When the results from Sourmash and BinDash are expressed as ANI percentages, they can be directly compared to the ANI percentages generated by alignment-based tools like PyANI and FastANI. This would provide a more unified framework for evaluating and comparing genome similarity tools, bridging the gap between alignment-based and alignment-free methods.

The study also assessed computational efficiency through average time per comparison. K-mer-based tools proved to be orders of magnitude faster, with average times in milliseconds, making them suitable for large-scale analyses where computational resources are a constraint.

These findings contribute to a nuanced understanding of genome comparison methodologies, offering practical guidance to researchers. Alignment-based methods remain the gold standard for precision, while k-mer-based tools provide scalable solutions for high-throughput analyses. Using these insights, this study supports the selection of appropriate tools based on the specific requirements of genomic research, advancing the field of comparative genomics.

Appendices

Appendix A

Appendices I

A.1 Code and Repository Information

The computational pipeline developed for this study, including the genome filtering program used to select the 175 genomes from the NCBI database, is publicly available on GitHub. This repository contains:

- Python scripts for the automated genome comparison pipeline.
- A filtering script for selecting genomes based on completeness from GTDB metadata.
- Usage instructions.

The repository can be accessed at the following link: https://github.com/chandranerella/genomeComparison.

A.2 Accession Numbers of the Genomes

This appendix lists the accession numbers for all 175 genomes used in this study.

Accession Number	Accession Number	Accession Number	Accession Number	Accession Number
GCA_000773965.1	GCF_008631595.1	GCA_002554555.1	GCF_008631615.1	GCA_002959495.1
GCA_002959555.1	GCF_008631635.1	GCA_002959725.1	GCF_008631645.1	GCA_003149005.1
GCA_003327525.1	GCF_008631665.1	GCA_003634335.1	GCF_008631685.1	GCA_003970755.1
GCA_004295855.1	GCF_008631715.1	GCA_009668235.1	GCF_008631725.1	GCA_010677205.1
GCA_016625125.1	GCF_008631735.1	GCA_016625195.1	GCF_008631795.1	GCA_016643525.1
GCA_019104105.1	GCF_008631815.1	GCA_900068845.1	GCF_008631825.1	GCA_900068865.1
GCF_000025405.2	GCF_008631835.1	GCF_000179655.1	GCF_008631865.1	GCF_000220605.1
GCF_000233595.1	GCF_008631895.1	GCF_000241285.1	GCF_008631905.1	GCF_000255315.1
GCF_000270125.2	GCF_008631925.1	GCF_000282675.1	GCF_008631935.1	GCF_000475035.1
GCF_000582575.1	GCF_008631975.1	GCF_000627115.1	GCF_008631995.1	GCF_000687245.1
GCF_000709995.2	GCF_008632005.1	GCF_000710015.2	GCF_008632095.1	GCF_000743785.2
GCF_000745295.1	GCF_009728735.1	GCF_000757415.2	GCF_009759885.1	GCF_000757435.2
GCF_000969395.1	GCF_009765475.1	GCF_001264395.1	GCF_009830025.1	GCF_001288285.1
GCF_001543055.1	GCF_009996785.1	GCF_001558735.2	GCF_010523255.1	GCF_001597625.1
GCF_001641135.1	GCF_010667965.1	GCF_001709315.1	GCF_011752685.1	GCF_002157425.2
GCF_002215245.1	GCF_012241415.1	GCF_002224585.2	GCF_013201565.1	GCF_002287745.1
GCF_002811195.1	GCF_013201575.1	GCF_002952035.2	GCF_013403175.1	GCF_002959435.1
GCF_003096515.1	GCF_013403215.1	GCF_003148935.1	GCF_013403225.1	GCF_003182175.1
GCF_003182295.1	GCF_014138175.1	GCF_003258435.1	GCF_014138195.1	GCF_003268795.1
GCF_003369485.1	GCF_014138205.1	GCF_003704305.1	GCF_014138245.1	GCF_003860325.1
GCF_004028255.1	GCF_014156615.1	GCF_004117135.1	GCF_014353705.1	GCF_004136415.1
GCF_004344785.1	GCF_014353835.1	GCF_004358685.1	GCF_014353845.1	GCF_004793995.1
GCF_005233495.1	GCF_014353865.1	GCF_005233795.1	GCF_014353935.1	GCF_005233805.1
GCF_006385265.1	GCF_014353945.1	GCF_006385385.1	GCF_014353955.1	GCF_007050955.1
GCF_007665685.1	GCF_014353995.1	GCF_007828605.1	GCF_014354035.1	GCF_008039815.1
GCF_008631275.1	GCF_014354135.1	GCF_008631285.1	GCF_014495885.1	GCF_008631295.1
GCF_008631335.1	GCF_014838905.1	GCF_008631345.1	GCF_014838975.1	GCF_008631375.1
GCF_008631415.1	GCF_014839015.1	GCF_008631445.1	GCF_014839085.1	GCF_008631475.1
GCF_008631505.1	GCF_014839115.1	GCF_008631515.1	GCF_014839125.1	GCF_008631555.1
GCF_008631565.1	GCF_014839205.1	GCF_008631595.1	GCF_014839265.1	GCF_008631615.1
GCF_008631635.1	GCF_014839315.1	GCF_008631645.1	GCF_014839325.1	GCF_008631665.1
GCF_008631685.1	GCF_014839425.1	GCF_008631715.1	GCF_014841465.1	GCF_008631725.1
GCF_008631735.1	GCF_015354545.1	GCF_008631795.1	GCF_016598655.1	GCF_008631815.1
GCF_900215395.1	GCF_900454395.1	GCF_900454505.1	GCF_900185875.1	GCF_902706165.1
GCF_018257655.1	GCF_019048385.11	GCF_900119405.1	GCF_900167295.1	GCF_900167425.1

Table A.1: Accession numbers of the 175 genomes used in this study

Bibliography

[1] L. Tian, R. Mazloom, L. S. Heath, and B. A. Vinatzer, "LINflow: a computational pipeline that combines an alignment-free with an alignment-based method to accelerate generation of similarity matrices for prokaryotic genomes," *PeerJ*, vol. 9, p. e10906, 2020.

[2] M. S. Waterman, "General methods of sequence comparison," *Bulletin of Mathematical Biology*, vol. 46, no. 4, pp. 473–500, 1984.

[3] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, pp. 403–410, Oct. 1990.

[4] D. Wheeler and M. Bhagwat, "BLAST QuickStart," in *Comparative Genomics: Volumes 1 and 2*, Humana Press, 2007.

[5] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[6] S.-H. Yoon, S.-m. Ha, J. Lim, S. Kwon, and J. Chun, "A large-scale evaluation of algorithms to calculate average nucleotide identity," *Antonie van Leeuwenhoek*, vol. 110, pp. 1281–1286, Oct. 2017.

[7] S. Ciufo, S. Kannan, S. Sharma, A. Badretdin, K. Clark, S. Turner, S. Brover, C. L. Schoch, A. Kimchi, and M. DiCuccio, "Using average nucleotide identity to improve taxonomic assignments in prokaryotic genomes at the NCBI," *International Journal of Systematic and Evolutionary Microbiology*, vol. 68, no. 7, pp. 2386–2392, 2018.

[8] J. Goris, K. T. Konstantinidis, J. A. Klappenbach, T. Coenye, P. Vandamme, and J. M. Tiedje, "DNA–DNA hybridization values and their relationship to whole-genome sequence similarities," *International Journal of Systematic and Evolutionary Microbiology*, vol. 57, no. 1, pp. 81–91, 2007.

[9] C. Jain, L. M. Rodriguez-R, A. M. Phillippy, K. T. Konstantinidis, and S. Aluru, "High throughput ANI analysis of prokaryotic genomes reveals clear species boundaries," *Nature Communications*, vol. 9, no. 1, p. 5114, 2018.

[10] I. Lee, Y. Ouk Kim, S.-C. Park, and J. Chun, "OrthoANI: An improved algorithm and software for calculating average nucleotide identity," *International Journal of Systematic and Evolutionary Microbiology*, vol. 66, no. 2, pp. 1100–1103, 2016.

[11] M. Palmer, E. T. Steenkamp, J. Blom, B. P. Hedlund, and S. N. Venter, "All ANIs are not created equal: implications for prokaryotic species boundaries and integration of ANIs into polyphasic taxonomy," *International Journal of Systematic and Evolutionary Microbiology*, vol. 70, no. 4, pp. 2937–2948, 2020.

[12] N. T. Pierce, L. Irber, T. Reiter, P. Brooks, and C. T. Brown, "Large-scale sequence comparisons with sourmash," *F1000Research*, vol. 8, p. 1006, 2019.

[13] R. Riedel, F. M. Commichau, D. Benndorf, R. Hertel, K. Holzer, L. E. Hoelzle, M. S. Y. Mardoukhi, L. E. Noack, and M. Martienssen, "Biodegradation of selected aminophosphonates by the bacterial isolate ochrobactrum sp. BTU1," *Microbiological Research*, vol. 280, p. 127600, 2024.

[14] J. Zhao, X. Zhao, J. Pierre-Both, and K. T. Konstantinidis, "BinDash 2.0: New MinHash scheme allows ultra-fast and accurate genome search and comparisons."

[15] B. D. Ondov, T. J. Treangen, P. Melsted, A. B. Mallonee, N. H. Bergman, S. Koren,

and A. M. Phillippy, "Mash: fast genome and metagenome distance estimation using MinHash," *Genome Biology*, vol. 17, no. 1, p. 132, 2016.

[16] L. Pritchard, "widdowquinn/pyani." https://github.com/widdowquinn/pyani, 2024. Accessed: 2024-09-08. Last updated: 2024-08-29.

[17] L. Irber, P. T. Brooks, T. Reiter, N. T. Pierce-Ward, M. R. Hera, D. Koslicki, and C. T. Brown, "Lightweight compositional analysis of metagenomes with FracMinHash and minimum metagenome covers."

[18] S. Kurtz, A. Phillippy, A. L. Delcher, M. Smoot, M. Shumway, C. Antonescu, and S. L. Salzberg, "Versatile and open software for comparing large genomes," *Genome Biology*, vol. 5, p. R12, Jan. 2004.