universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

"Improvement of Plasmid Prediction in Metagenomic Sequencing Data"

verfasst von / submitted by

Michael Predl BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2020 / Vienna, 2020

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 066 875
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Master Bioinformatik
Betreut von / Supervisor:	Univ.-Prof. Mag. Dr. Thomas Rattei

Contents

Kurzfassung

Plasmide sind ein wichtiger Untersuchungsgegenstand der modernen Mikrobiologie, da sie eine schnelle und horizontale Verbreitung von Genmaterial in mikrobiellen Gemeinschaften bewerkstelligen. Weiters ist der Inhalt der Plasmide von großer Bedeutung in der Ökologie und Medizin, da sie ihren Wirten Gene unter anderem für Umwelt- und Antibiotikaresistenzen, Verarbeitung von Metaboliten, sowie Virulenz bereitstellen. Sie werden in der Biotechnologie bereits ausgiebig genutzt, in Umweltproben, vor allem in Metagenomen, sind sie jedoch kaum untersucht.

Weiters gibt es durch die schnelleren und günstigeren Sequenziermethoden immer mehr (metagenomische) sequenzierte Proben, von denen viele jedoch nur auf einzelne Aspekte untersucht wurden, wie zum Beispiel die phylogenetische Einordnung der in der Probe befindlichen Organismen oder die Extraktion der Genome bislang unbekannter Mikroorganismen. Die Frage ob und welche Plasmide sich in einer Probe befinden bleibt meist außen vor und unbeantwortet.

Trotz, oder gerade wegen dieser Unterrepresentation von Plasmiden in Metagenomen sind in den letzten Jahren einige bioinformatische Werkzeuge publiziert worden die sich mit den verschiedensten Zugängen dem Problem widmen, Plasmidsequequenzen aus Metagenomen zu extrahieren. Diese Masterarbeit testet die momentan verfügbaren Programme zur Plasmidvorhersage mittels drei simulierten metagenomischen Datensätzen. Durch diese werden die Grenzen derzeitiger Methoden aufgezeigt und die verschiedenen Ansätze verglichen.

Aufgrund dieser Erkenntnisse und aufbauend auf Ideen und Konzepte von zwei bestehenden Programmen, Recycler und MetaPlasmidSPAdes, wird CAPP, ein neues Programm zur Vorhersage von Plasmiden in metagenomischen Datensätzen, entwickelt. Im Kontext der drei simulierten Datensätze erreicht CAPP dadurch eine größere Anzahl korrekt rekonstruierter Plasmide, als seine Vorgänger.

Abstract

Plasmids are an important subject in modern microbiology, as the convey fast and horizontal transfer of genetic material in microbial communities. Further, the content of plasmids is significant for the fields of ecology and medicine, as they provide their hosts with resistance to antibiotics and environmental stress, processing of metabolites, as well as virulence. While they are extensively used in biotechnology, they are hardly studied in environmental samples, especially metagenomes.

Further, the faster and less expensive sequencing methods lead to more (metagenomic) sequencing samples, most of which are only analyzed in specific aspects, as taxonomic classification of organisms within the sample, or the extraction of genomes of so far unknown microorganisms. The question whether or what plasmids are present in the sample often remains unanswered.

In spite of, or due to this under-representation of plasmids in metagenomes, a variety of bioinformatics tools has been published in recent years, targeting the extraction of plasmid sequences from metagenomes. This master's thesis evaluates the currently available plamid prediction programs using three simulated metagenomic sequencing datasets, uncovering the limitations of these state of the art methods and their approaches.

Based on these findings and improving upon the ideas and concepts of two existing methods, Recycler and MetaPlasmidSPAdes, a new program for plasmid prediction in metagenomic sequencing data, called CAPP, is developed. In the context of the three simulated samples, CAPP achieves a higher number of correctly reconstructed plasmids, than its predecessor.

Abbreviations

- FN - false negative
- FP - false positive
- HGT - horizontal gene transfer
- kb - kilobase, 1000 nucleobases of DNA or RNA
- MPS - MetaPlasmidSPAdes
- NCBI - National Center for Biotechnology Information
- oriT - origin of transfer
- oriV - origin of vegetative replication
- TN - true negative
- TP - true positive
- T4CP - type four secretion coupling protein
- T4SS - type four secretion system
- WGS - whole genome sequencing

1 Introduction

Plasmids are extrachromosomal DNA elements, present in bacteria, archaea, but also eukaryotes.[43, 49] They convey functionality including virulence, resistance to antibiotics and environmental adaptation.[22] Plasmids replicate independently of the host and can be transferred across different organisms and species. Thus, the study of plasmids is important in the study of microbial communities and their ecology, but also human disease and the spread of antibiotic resistance.

Plasmdids are also heavily used in biotechnology, as they can easily be altered, transferred and amplified. Among other uses, they facilitate the production of proteins of interest, the amplification of target DNA and allow for the addition of genes or genetic curcuits to a bacterial genome.

The detection of plasmids is challenging, as no universal marker exists. Thus, tedious extraction and cultivation are often used to study and identify novel plasmids.[44] With the advances in sequencing technologies, whole genome sequencing (WGS) of entire metagenomes are possible. These metagenomic samples include plasmid DNA, which can be potentially be used to study plasmids in the natural environment and community, with reduced cost and time requirements. Multiple approaches for plasmid prediction in metagenomic sequencing samples have been implemented, ranging from reference sequence based to machine learning. Despite these efforts, the correct reconstruction of plasmid remains a challenge in short read sequencing samples.[24, 5]

The starting point for this thesis and the improvement of plasmid prediction methods is a report of colleagues about Recycler[41], a plasmid prediction tool, requiring a lot of compute time on some of their samples. Therefore, the initial goal is the assessment of current plasmid prediction methods and the improvement upon Recycler, to produce meaningful predictions in a timely manner.

2 Background

This section provides background information necessary for the understanding of the remaining thesis. First, a detailed overview of the biology and relevance of plasmids will be given (section 2.1). Next, concepts and steps of a metagenomic sequencing experiment are explained, with a focus on the information needed for the methods in plasmid prediction. This covers the field of metagenomics, the technical backgrounds of DNA sequencing technologies, together with information on sequence assembly and generation of simulated sequencing data (sections 2.2.1 to 2.2.4). In the last part of the background information, current plasmid prediction are listed and the key ideas of their approach are explained (section 2.3).

2.1 Plasmids

Plasmids are extrachromosomal DNA elements, replicating independent of the chromosomal DNA. While the majority of known plasmids are found in bacteria, they are found in archaea and eukaryotes as well. Plasmids show a wide range of GC content and size[43], with very large plasmids of more than 100kb being found as well.[17, 22] Their shape is usually circular, but linear plasmids have been found in a variety of species as well.[12]

Plasmids often carry benefitial genes, which help the host cell to adapt to different environments, as antibiotic resistance, heavy metal resistance or virulence. However, small plasmids with essential genes, as the ribosomal RNA genes, have been identified as well.[2]

Further, very large DNA molecules, carrying essential genes, but also a plasmid-like replication machinery, have been found. As these large molecules have characteristics of chromosomes and plasmids alike, they have been termed chromids.[17]

Due to their independent replication, plasmids are often present in more than a single copy per cell and plasmid copy numbers can range from a single plasmid in a cell up to several hundred.[19, 20, 53, 10] The amount of plasmid DNA can even exceed the amount of chromosomal DNA.[53] While in some species, plasmid copy number is proportional to the size of the plasmid[53], several other factors influence the copy number as well. The primary genetic factor in plasmids is the origin of vegetative replication (oriV), including regulatory elements. These elements have altered and selected to achieve high plasmid copy numbers for the use in expression vectors or cloning. But also environmental factors or community structure as biofilm formation have an impact on the plasmid count per host cell.[10] Further, the copy number of plasmids has an impact on the fitness of both the plasmid and the host cell, and is therefore subject to different forces of selection.[50]

Many plasmids are not only inherited to the offspring, but can be transferred horizontally to different organisms. Given this mobility, plasmids are a huge factor in horizontal gene transfer (HGT)[1] and have been shown to be transmitted over large geographic distances in the ocean and shared across different species.[36]

This transmission can happen by transformation, which is the uptake of free DNA from the environment, or by conjugation, an active process of DNA transfer between two cells. Conjugation requires a set of factors, however, only conjugation by the type four secretion system (T4SS) is well studied.[43] This type of conjugation depends on an origin of transfer (oriT) on the plasmid, a relaxase protein, a type four secretion coupling protein (T4CP) and and the proteins of the T4SS machinery.

Plasmids can be classified by their mobility, as they can be self-transmissive, if they carry all genes needed for their conjugation, mobilizable, if they do not

carry all the necessary genes, but can be transferred if the missing genes are encoded on another molecule in the same cell, or non-mobilizable, if they do not have any of the necessary genes for conjugation.

Several other approaches to classify plasmids more have been published and target different elements of plasmids as the replication machinery, mobility related genes as the relaxase and mating pair formation related proteins. While these definitions show the variability of plasmids, they are all dependent on the understanding of the mechanism of conjugation or replication and can therefore not be applied to all plasmids.[34]

While plasmids are common and known for a long time, their extraction is still challenging.[44] Whereas historically most plasmids were characterized from isolates, this approach can only cover a small portion of organisms. Approaches like metagenomic plasmidome isolation are able to identify plasmids beyond these limitations.[22, 12] Another technique, physically linking DNA molecules and retaining this proximity information throughout the sequencing experiment can help disentangle the metagenome and assign plasmids to their hosts.[7] The drawback of these methods is the need for specialised and more complex experimental protocols, which require trained personal and come with increased time and cost. Thus, the prediction of plasmids out of whole genome sequencing metagenomic samples would be preferrable, especially with sequenced metagenomes available in public databases.[45] However, as short read sequencing technologies are dominant in metagenomics, the recovery and prediction of plasmids from these samples is difficult. Therefore many different plasmid prediction methods using a variety of approaches have been developed for this type of sequencing data and will be reviewed in this thesis.

2.2 Metagenomic sequencing

2.2.1 Metagenomics

The term metagenome goes back to Handelsman et al. in 1998[16] and describes the entirety of genomic material of all organisms within a community or an environment. Since then, metagenomics has turned into its own field in biology, as it gives researchers the opportunity to study organisms that cannot be cultured so far.[47] A variety of environments has been studied, ranging from aquatic and soil samples to the human gut microbiome. Advances in metagenomics allow even for the extraction and reconstruction of novel, near complete genomes.[31] Metagenomic sequencing experiments typically follow the steps of DNA extraction, sequencing, assembly and downstream analysis. This thesis is focused on the last two steps, assembly and the downstream analysis, namely plasmid prediction. Thus, one should be aware of biases and errors in the data at the point of the assembly that have originated from the previous steps.

2.2.2 DNA extraction and sequencing

DNA extraction has been shown to have varying results depending on the organism. However, this bias mostly affects the efficiency of DNA extraction, leading to a skewed reflection of relative abundance or even the complete failure to detect certain species.[51]

For the next step, the sequencing of the exracted DNA, different technologies can be used. While promising new technologies resulting in long contiguous reads are available and have already been used in whole genome sequencing of metagenomes, due to sequencing costs, the dominant sequencing technology in metagenomics is the Illumina technology. This sequencing technology uses bridge amplification to generate clusters of identical dna fragments. Due to this am-

plification method, the fragment maximum fragment length is 600bp to 700bp. Similarily, the read length ranges from 100bp to 300bp, depending on the type of Illumina sequencing machine.[13] When exceeding this length, read quality gets worse and the base calls cannot be trusted.

Apart from the restrictions in fragment and read length, the sequencing process is biased in its errors or success rate by the properties of the DNA itself, such as the GC content.[9] Additionally, insertion, deletion and substitution errors specific to the sequencing technology itself occur.

Illumina sequencing has the option for paired end sequencing. In this process, both strands of the fragment are sequenced starting at either end of the molecule. Read pairs provide more information than two single reads, as they are known to be from the same fragment and thus, their ends have to be within a certain range according to the fragment length. Further, if the sum of the two read lengths exceeds the length of the fragment, the two reads overlap and can be joined to act as a longer read using bioinformatics tools.

While correct sequencing cannot be expected, the error rate for Illumina reads is typically in the range of less than 1% per base.[37] Additionally, an estimate of the correctness of a base call is given by the PHRED-score along with the sequenced read. This score can be used in quality control to trim and filter sequencing reads that are likely to be incorrect.

Additionaly, variability in the sequencing data originates from the DNA preparation for the sequencing process. In the case of Illumina sequencing, the extracted DNA is fragmented to allow for efficient amplification. This fragmentation leads to a distribution of differently sized DNA fragments, rather than an exact length. Another variability in Illumina sequencing originates from the fact that only a subset of DNA molecules is actually sequenced. This subset is a random sample of the DNA fragments, which leads to some stretches of DNA being represented

in multiple reads, while others may not be sequenced at all, due to chance. Underrepresented or missing DNA stretches can then hinder the correct assembly of the sequences. To counter this effect, one can sequence more material, increasing sequencing depth, leading to higher coverage, meaning more reads covering the same stretch of DNA. In the setting of metagenomics, many communities include organisms of low abundance, which, in order to reach sufficient coverage, require extensive sequencing depth. As the community composition is in most cases not known prior to sequencing and due to the cost of high sequencing depth, in metagenomic sequencing experiments it can be expected that some genomes will be poorly covered, incomplete or not captured at all.

2.2.3 Simulated sequencing data

The simulation of sequencing experiments for benchmarking bioinformatics tools is a common strategy and a large variety of methods for simulation exists.[42, 13] These tools cover several sequencing technologies and error types, but despite these efforts produce data with less complexity, bias and errors than real sequencing experiments. In addition to the technical aspect of sequencing simulation comes the simulation of the biological sample. In the case of metagenomics, a sample does not only consist of a set of organisms of different abundance, but also strain variety and random mutations within the same species, contamination through different sources and mobile elements as viruses and plasmids. Thus, a simulated sequencing sample cannot be expected to represent a potential real sample, but it allows for a testing ground with complete control over a variety of parameters and knowledge of the complete composition of the community of organisms.[28]

2.2.4 Sequence assembly

Sequence assembly is the transformation of the many individual reads into larger sequences. The aim of an assembly is to reconstruct the correct DNA sequences and reduce fragmentation. Different approaches for assembly of sequencing data have been developed, but the most prominent method for short read assembly is the DeBruijn graph.[30]

In this approach, all k-mers, a stretch of DNA with length k, are counted in the sequenced reads. Then, a directed graph is constructed with the nodes being the set of k-mers and edges are drawn between all nodes that overlap by $k-1$ bases. The individual reads are not directly represented in the graph, but rather by the count of occurrences of the k-mers, which can be translated to the coverage of that stretch of DNA. Assemblers using this strategy then connect nodes that are unambiguous, meaning a node with only one outgoing edge which is the only incoming edge of the second node. While this already reduces the number of nodes and edges of the graph and produces long sequences, assemblers can make use of the read information and coverage to remove edges that are not reflected in the reads and remove potential sequencing errors. Additionally, read pairs can be used to identify correct paths through the graph.[39]

The resulting assembly graph contains a set of connected contiguous stretches of DNA (contigs) of variable length, with assigned coverage. Completely resolving the assembly graph is often not possible, as shared and repetitive sequences of size greater than read or fragment length cannot be resolved. Further, incomplete sequencing leads to gaps in the assembly graph, that cannot be closed. Lastly, assembly is not without errors and is ultimately dependent on the size of the k-mers. A small k leads to many small fragments that are highly connected, while large k leads to larger, but often disconnected contigs.[30]

While these are the basics of DeBruijn graph based assembly, many specialized

methods have been developed using a variety of strategies and with distinct aims. For example, the assembly of metagenomes requires different strategies than the assembly of a genome of a bacterial isolate.[6, 32]

Despite the improvements and successes in sequence assembly, the output of assemblers varies largely in correctness, completness and contiguity across different methods and samples.[42, 48] A recent study shows that the choice of the assembler can impact the performance of plasmid prediction as well.[15]

In metagenomics, several methods have been developed to validate assemblies and correcting errors.[33] There are however, no approaches on the validation of plasmid predictions, leading to attempts of validation, refinement or error correction by running a different plasmid prediction tool on the same dataset or the prediction, rather than having an explicit validation method.[4, 35]

2.3 Current plasmid prediction methods

In this section, an overview over all the different state of the art plasmid prediction tools will be given. While the approaches are different for all the methods, they share some common features. First of all, prediction starts always at the assembly level, regardless of the assembly happening internally, as in PlasmidSPAdes and MetaPlasmidSPAdes, or being required as a data preprocessing step.

Further, the methods can be grouped into three categories, the assembly graph based algorithms, marker based search, and the machine learning classifiers. As the name suggests, assembly graph based algorithms use the assembly graph as a means to reconstruct whole plasmids, by utilizing the connection of different sequences. Marker based search methods utilize databases of different plasmid specific marker genes or sequences to identify plasmid sequences. Machine learning classifiers on the other hand, only use the pure sequence information

of contigs, and classify their origin as plasmid or chromosome using a classifier previously trained on a set of known plasmid and chromosome sequences.
While all groups try to extract plasmids form a metagenomic sequencing samples, two different goals of plasmid prediction can be formulated. First, the reconstruction of complete plasmid sequences, grouped to separate individual plasmids. Second, the classification of every contig in the assembly according to its origin. As these aims are different they will be evaluated separately in this thesis.

2.3.1 Recycler

Recycler[41] is an assembly graph based method with the approach of identifying circular segments in the assembly graph. In addition to the assembly, a bam file containing the sequencing reads mapped to the nodes of the assembly graph is needed as input. To ensure that the selected cyclic sequences are plasmids, they are tested for uniform coverage and paired end information coherent with the proposed path. To disentangle overlapping plasmids, accepted solutions are pulled off the graph and node coverage is reduced accordingly.

2.3.2 MOB-Suite

MOB-Suite[40] is a conglomerate of methods for the reconstruction and typing of plasmids. The method for the prediction of plasmids from an assembly is called MOB-recon. Besides the contigs of the assembly, a database of plasmid sequences and markers is required. While the authors provided such a database with the release of MOB-Suite, a custom database can be used. MOB-recon identifies likely plasmid sequences by database search and then groups these contigs based on similarity to known plasmid clusters. By this approach, theoretically, novel plasmids with replicon or relaxase markers similar to already known ones, can

be correctly identified.

2.3.3 PlasmidSPAdes

PlasmidSPAdes[3] is a method which includes assembly and plasmid prediction, and is part of the SPAdes package. The rationale of this method is based on the assumption of the sample being a sequenced bacterial isolate which may contain plasmids. In this setting, the majority of contigs is expected to belong to the chromosome and show uniform coverage. As plasmids are in most cases present in a higher copy number than the chromosome, plasmid contigs can be expected to be sequenced with higher depth. Therefore, PlasmidSPAdes tries to identify the coverage level of the chromosomal sequences and applies a coverage cutoff removing all nodes likely to belong to the chromosome. The remaining graph is analyzed for circular elements and connected components exceeding a size threshold, which are then presented as putative plasmids.

2.3.4 MetaPlasmidSPAdes

MetaPlasmidSPAdes[4] is again a method beloging to the SPAdes package. As PlasmidSPAdes is conceptually tied to isolate sequencing samples, a new method was developed for metagenomic datasets. The idea of applying a coverage cutoff for graph simplification and plasmid enrichment is retained. As chromosomal sequences cannot be identified by coverage, MetaPlasmidSPAdes applies a hard coverage cutoff and removes any contiguous elements of the assembly graph below the threshold. The remaining graph is searched for cyclocontigs, circular paths in the assembly graph, without any branches, and plasmid-like connected components. To detect plasmids of high coverage that are overlapping with lower coverage sequences, the cutoff is increased iteratively, until it exceeds the maximum coverage in the graph.

2.3.5 PlasFlow

PlasFlow[23] is a machine learning approach using kmer information to classify the contigs of an assembled sample. The classifier is trained on a set of known plasmid and chromosome sequences, as well as their kmer frequencies of size $k = 5$, 6 and 7. The three different sizes of k are each used in an individual classifier, with PlasFlow combining the results of all three by majority vote. In addition to plasmid or chromosome origin, PlasFlow predicts the taxonomy of a contig on phylum level as well. The two different classification types are treated independently, leading to contigs being classified in both origin and taxonomy, only one of the categories, or completely unclassified.

2.3.6 PlasClass

PlasClass[35] is a machine learning approach building upon the idea of PlasFlow. As PlasFlow, PlasClass uses kmer frequencies for classification of contigs, but the contigs are divided into different classes by size. For each category of contig size, a different classifier is trained and used. The output of PlasClass is a list of contig names and a score between 0 and 1 reflecting the classification of the origin of the sequence being chromosome or plasmid.

2.3.7 PlaScope

PlaScope[11] is a contig classification method aiming to identify plasmids based on a database of already known plasmids and chromosomes. The classifier within PlaScope is centrifuge[21], a tool for the classification of metagenomic contigs based on exact sequence matches to a database. In PlaScope, the categories for classification are set to plasmid, chromosome and unclassified, with each contig being assign to one of these categories. Kim et al. published two different reference databases that can be readily used, with a focus on *E. coli* and *Klebsiella*

respectively, beyond a common set of known plasmids of different species. In addition to these databases, custom databases can be created and used.

2.3.8 Excluded methods

For completeness sake, the tools Placnet[25], PlasmidFinder[8] and cBar[54] should be mentioned as existing plasmid prediction tools. These were excluded from the evaluation in this thesis, due to a variety of reasons.

To start with, cBar is a machine learning approach which classifies contigs based on kmer composition as plasmid or chromosome. The program could however, never be tested, as neither the source code, nor a compiled version were available online.

Placnet is a pipeline to include various information as plasmid markers and sequence similarity to known plasmids in the assembly graph. But, as it requires manual curation to classify sequences as plasmid, no objective evaluation was possible in this thesis.

PlasmidFinder is a tool for identifying plasmids by searching for known plasmid marker genes. While this is a valid approach for identifying plasmid sequences, the output of PlasmidFinder cannot be used for evaluation in this context, as it only states the marker gene and its location, but does not aim to reconstruct whole plasmids or classify all contigs.

3 Problem description and research goals

The extraction or reconstruction of plasmids from a metagenomic sequencing sample is an open topic. There exist several distinct approaches and just as many different variations of the goal of plasmid prediction. Thus, it is needed to specify and define the problem to be tackled more exactly, in order to verbalize concrete goals.

The first aim is to extract individual plasmids, as this enables research on plasmids as individual entities, as evolution and propagation, as well as being helpful for experimental validation.

Additionally, I will focus on finding novel plasmids. This means that plasmids need to be identified by their inherent properties, but not by similarity to already known sequences.

With the great differences in sequencing technologies and their data characteristics, it is necessary, in the scope of this master thesis, to limit the problem to a single type of sequencing technology. In this thesis, I will focus on short paired end reads, as produced by Illumina sequencers, since this type of sequencing data is very common in the field of WGS metagenomics.

The aim of this thesis is to improve plasmid prediction in the before mentioned scope of the problem. This includes a performance benchmark of all state of the art plasmid prediction tools. Further, programs and algorithms that are coherent with the narrowed scope of plasmid prediction of this study will be evaluated for their limitations and flaws, in order to improve upon them. Using this information, the goal is to develop a plasmid prediction tool with increased predictive power.

4 Methods

4.1 Evaluation

In this section, all methods regarding the generation and evaluation of plasmid prediction output will be described.

4.1.1 Benchmark design

The benchmark consists of three different sequencing read samples. All of the samples have the same number of reads and are simulated using the same parameters. The difference of the benchmark samples is in their complexity, their number of species and shape of the abundance distribution. The three benchmark samples consist of 20, 50 and 200 species respectively and will be referred to by that number. The shape of the abundance distribution is modeled as a power law distribution as suggested by Pignatelli et al.[38]

$$(1 + s * r) * i^b \tag{1}$$

A random factor $(1 + s * r)$ is included, to account for some of the natural variability. This factor is composed of a random number r between 0 and 1 and a scaling factor s, determining the impact of the random factor.
The slope of the power law distribution is determined by the variable b (equation 1, figure 1), a shallow slope increases the complexity of the metagenomic analysis, as the differences in abundance of the species and coverage of their sequences are low. As the three samples represent increasing levels of complexity, the variable b is set to -1.25, -0.75 and -0.25 for the 20, 50 and 200 species samples respectively.

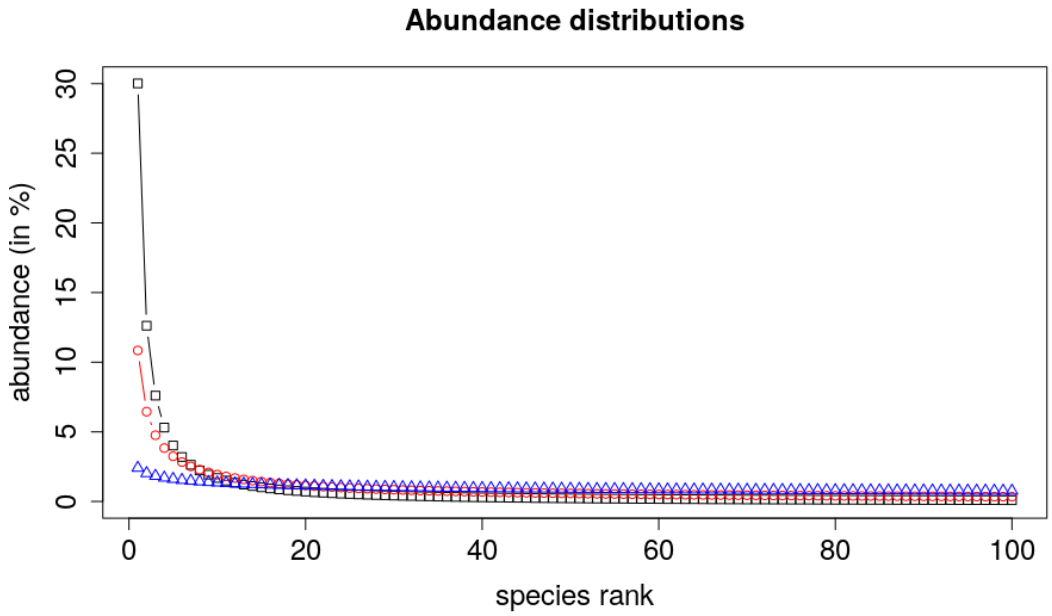


Figure 1: Abundance distributions with different values for the variable b in the power low function (black: -1.25, red: -0.75, blue: -0.25). The abundance is shown as percentage of a community of size 100.

4.1.2 Sample and community design

For the benchmark samples, the list of all bacterial and archaeal species with a complete assembly in NCBI RefSeq is used. Of those prokaryotes, only those with at least one plasmid in the assembly are considered for the benchmark. The selection which species are used in the samples is random, but at least one representative of each of the large clades of bacteria and archaea is selected, to counter the over-representation of model organisms and human pathogens in the database. The selected organisms for each of the samples can be found in supplemental table S1. A python script (genome_list_parser.py) is used to automatize the selection step. It uses as input the tab delimited file of the NCBI prokaryote database, that was manually filtered to contain only complete assemblies that include at least one plasmid. The output of the script is a filtered version of the

input file, containing only the header and the selected species.

A script was written to open the assembly fasta file for each selected species of a sample, extract the identifiers of the different sequences as well as their classification (plasmid or chromosome) and write them as individual fasta files to a new directory. Additionally, a text file is generated containing the species name and all the chromosomes and plasmids in the assembly. This file is then used to assign each species and reference sequence an abundance and the number of reads to be simulated.

In order to assign the number of reads to be drawn to each of the reference sequences, a relative abundance is assigned to each species. The species are put in random order and then assigned an abundance score based on their rank according to the power law distribution. The scores are then normalized to a sum of 1. Each of the species contains at least one chromosome and at least one plasmid. Now the copy numbers are assigned to these molecules: chromosomes get a value of 1, plasmids a value of 2 to the power of a random value of a normal distribution (equation 2, figure 2). The abundance assigned to the species is now split up and assigned to the different reference sequences proportional to their copy number. The read number is determined, using the abundance of each reference sequence, multiplied with their length, normalized over all references, as a probability that a read originates from a given reference sequence. Now for each reference sequence a binomial distribution with the remaining number of read pairs to be assigned as variable n (number of independent tests) and p being the probability of this reference being the origin of a read. The total number of read pairs that will be simulated for this reference sequence is drawn randomly from said distribution, to mimic the random sampling in a metagenomic sequencing experiment. After this process, the species and the number of reads to be simulated for each of their chromosomes and plasmids is determined. To ensure that

the total number of reads is assigned, the last assignment does not depend on a binomial distribution, but automatically assigns the remaining number of reads.

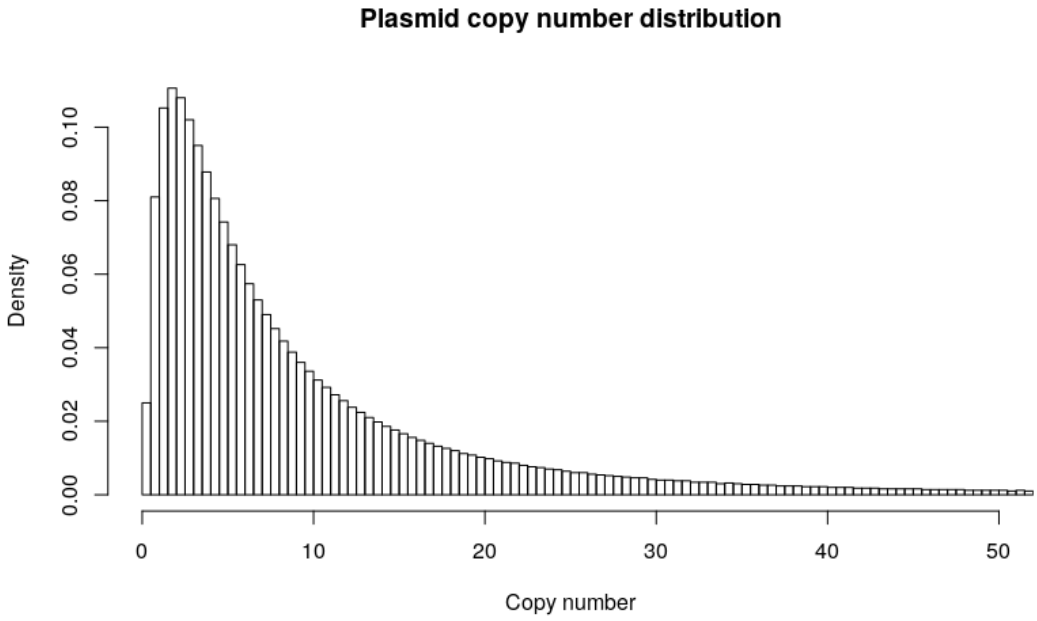


Figure 2: Histogram of the plasmid copy number distribution (10000 sample points, 0.001 to 0.999, see equation 2). Each bar has a width of 0.5.

$$2^{\mathcal{N}(\mu=\log_2(6),\sigma=1.6)} \tag{2}$$

This process of assigning the abundance and read number, is handled by a python script (ART_manager.py), which also handles the sequencing simulation. It takes as input a text file containing the species, plasmids and chromosomes, assigns abundance and read numbers as explained before and writes both to individual tab delimited files.

4.1.3 Sequencing simulation

For the sequencing simulation the ART sequence simulator (version "Mount-Rainier-2016-06-05")[18] is used. This version was modified to allow for circular reference sequences. The modification allows fragments to start at any position, in the case of the fragment exceeding ends of the reference sequence, the sequence of the other end is concatenated (figure 3, section 5.1.1).
For each sample 5 million read pairs are generated. The reads are simulated to be sequenced by an Illumina MiSeq v1 machine, with a readlength of 150 bp, a fragment length of 500 bp with a standard deviation of 25 bp. The sequencing simulation is done separately for each of the reference sequences and their fastq files are concatenated to retrieve the final fastq file for the sample.
The sequencing simulation of the hundreds of references in a sample is handled by the ART_manager.py script. Using either present abundance and read number files, or generating them de novo, the script calls the ART sequencing simulator for each reference file with all parameters as reference file location, name, read number, error profile, read length, fragment length and standard deviation. This can also be done in parallel, with the individual jobs in descending order of read number, to reduce the time needed. Apart from the sequencing simulation, a summary file is generated, containing the shape (circular or linear), length, abundance, read number and coverage of each reference sequence in a tab delimited list.

4.1.4 Assembly

For the assembly of the raw sequencing data, SPAdes[6] and MetaSPAdes[32] (both version 3.13.0)are used. The command line arguments are pe1-1 and pe1-2 for the forward and reverse sequencing reads and –meta is used to refer to MetapSPAdes. Otherwise, the program is run with default parameters.

No trimming or preprocessing is used for simulated sequencing reads.

4.1.5 Read mapping

Mapping the reads back to the contigs of an assembly graph is done using the same approach as described by Rozov et al.[41] The first step is to create a fasta file containing all contig sequences using the fastg file of the assembly. This is done using either the script provided in the Recycler installation, or a custom script used for CAPP. The script for CAPP extracts the contigs ID from a SPAdes fastg file and uses this numeric ID as fasta header. It also uses always the '+' strand of the contig, to allow for identification of the orientation of the reads mapped to the contigs. The fasta file is then used as input for BWA (version 0.7.16)[26] to map the reads to the contigs. The resulting output is then processed by samtools (version 1.9)[27] to create a bam file containing only primary alignments of the reads, that is sorted by the query, the contigs, and indexed.

4.1.6 Plasmid prediction

Plasmid prediction was carried out by 7 different published programs (Recycler, PlasmidSPAdes, MetaPlasmidSPAdes, MOB-Suite, PlaScope, Plasflow, PlasClass). Three additional tools have been found in the literature, but were excluded form the benchmark for different reasons. The first program, cBar[54], was not available on the link provided in the paper and could not be found within reasonable time of internet search. The second tool, PLACNET[25], requires manual curation and is thus not available for objective automated benchmarking. Plasmidfinder[8] predicts plasmid markers on contig sequences, but does neither aim to classify all contigs or to reconstruct complete plasmids. Thus, the output of Plasmidfinder cannot be included in the benchmark. PlasmidSPAdes (version

3.13.0) and MetaPlasmidSPAdes (version 3.13.0) are called as described in the assembly, with the respective command line flags to specify the programs. All other programs need an assembly graph as input. Recycler additionally needs a bam file containing the reads mapped to the contigs and the kmer size of the assembly. PlaScope, Plasflow and MOB-Suite all use the databases provided by their current version. In the case of PlaScope two different databases were available, here the "E. coli" database was selected over the "Klebsiella" database. To decrease the memory use of Plasflow a minimum size cutoff of 1kb was used for all contigs, as suggested by the authors on the github webpage of PlasFlow. To identify plasmids with MOB-Suite, the mob_recon program is used.

4.1.7 Evaluation of plasmid prediction on simulated data

To evaluate the predictions of the plasmid prediction programs, their output fasta files are aligned to the reference sequences of their simulated sequencing sample. This is done using nucmer (version 4.0.0) and the alignment processing tools of the MUMMER package, delta-filter and show-coords.[29] Delta-filter is used to restrict each base in the alignment of the query, the plasmid predictions, to be aligned only once, with the best hit in the reference sequences being selected. This is achieved using the -q flag. Show-coords creates a tab delimited text file from the alignment file of delta-filter. Here the options to show the length of the reference and query sequence (-l), the percentage of each sequence covered by the alignment (-c) and whether the query is terminal, identical or contained (-o) is chosen. Additionally, the list of alignments is sorted by the reference sequence (-r). The same procedure is also done to classify the contigs of the assembly, by using a fasta file of all contigs as query for nucmer.
With the predictions being classified and aligned to reference, the predictions are evaluated using several different metrics, aiming at correct and complete

plasmid reconstruction, individual plasmid binning and separation of plasmid and chromosome sequences. Each of the categories can be assigned a precision, sensitivity and F1 score, as calculated in equations 3 to 5.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{4}$$

$$F1 = 2 * \frac{Precision * Sensitivity}{Precision + Sensitivity} \tag{5}$$

In the category of separation of chromosomal and plasmid sequences, true positive is defined as the number of bases that are predicted to be plasmidic and are aligned to a plasmid reference sequence. In other words, the positives are sequences predicted to be of plasmid origin, negatives are sequences predicted to be chromosomal. The assignment of false and true positive or negative is done by comparing the prediction to the classification of the reference sequence it is aligned to. As only Plasflow, PlaScope and MOB-Suite predict sequences to be chromosomal, the proportion of false negative bases is calculated to be equal to the total length of plasmid reference sequences minus the bases that are true positives.

The definition is different for individual plasmid binning, where a bin is considered to be correct, if at least 80% of a plasmid reference sequence are part of a single prediction and at least 80% of the sequence in the prediction is aligned to this plasmid reference. True positives fulfill this criterium, false positives are all predicted bins that do not fulfill the criterium and false negative is the set of plasmids that is not successfully predicted. That way a meaningful precision and sensitivity, as well as an F1 score can be calculated for plasmid binning approaches. It should be noted that this category is not applicable to Plasmidfinder, Plasflow and PlaScope, as they do not separate sequences into

individual plasmids.

Further evaluation is done by restricting the space of evaluation to a subset of plasmids, eg. all plasmids that are not part of a certain database. Here, the true positives and false negatives are reduced, but the false positives are left untouched.

All of the evaluation is handled by a custom python script, that uses the information of the nucmer alignment file, as well as the output of the prediction programs to generate a text file showing the metrics mentioned before, as well as additional information on the individual predicted bins or sequences, like the reference sequence that the majority of bases is aligned to, the number of different total and chromosomal references that were aligned to it.

4.1.8 Assembly graph visualization

The visualisation of assembly graphs is done with Bandage (version 0.8.1).[52] It is a tool that allows for visualisation of assembly graphs with both strands combined or separated. It also makes custom colorization of contigs possible by combining a color tag with the contig name. Using nucmer to assign contigs of the assembly of each benchmark sample to their references, a file that colorizes each reference individually.

5 Results

In this section the primary results of this study will be presented. In the first section (5.1) the creation of simulated metagenomic sequencing samples and their characteristics is presented. The next section (5.2) shows some of the differences in predictive power between state of the art plasmid prediction tools, as well as examples of their shortcomings.
Section 5.3 revolves around CAPP, the novel plasmid prediction program that is developed for this thesis. The main ideas of improvement on existing tools and their algorithms are explained and the workflow of the novel program is presented. The main algorithms and improvements are explained and tested in detail in the sections 5.3.3 to 5.3.9. These sections present explanations and performance results of read mapping, the algorithm for cycle detection and scoring, the solution pull-off strategy, the performance improvements, as well as the candidate evaluation and classification.
The last part of this chapter revolves around the in-depth benchmark and comparison of all state of the art plasmid prediction tools. A detailed overview of the predictive power is given in section 5.4.1, including the differences in the choice of the assembler as well as the performance of the newly developed CAPP. In addition to the predictive performance, the computational load is analyzed as well and presented in section 5.4.2.

5.1 Generation of simulated sequencing data

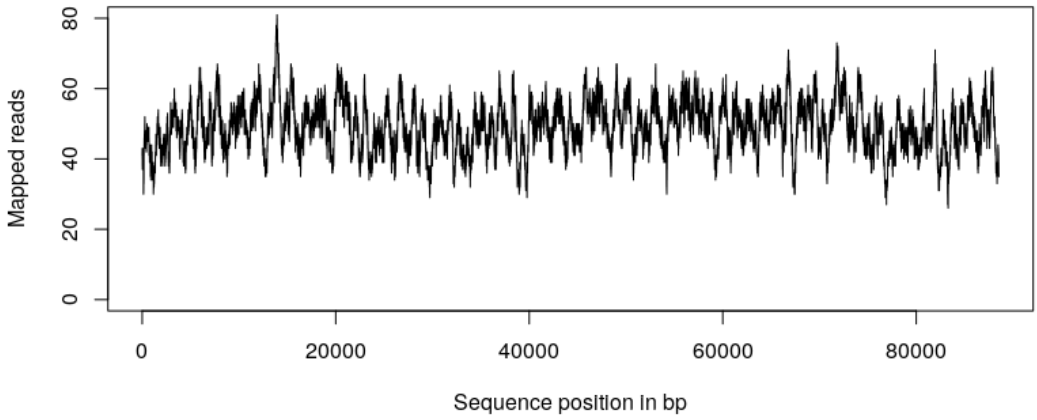
While a lot of comparisons and testing has been done for plasmid prediction tools, they often lack some of the current tools, only work on plasmidome data[41], treat all sequences as linear, or only use real data with a common truth strategy.[5, 24] While real data shows the whole complexity of sequencing

data, there is no metagenomic dataset containing the complete abundance and sequence composition of all species including their plasmids. This means that whatever is found by plasmid prediction programs needs to be experimentally validated, before being able to make any conclusions whether a prediction is correct or not. Therefore a simulated metagenomic dataset is needed, containing both chromosomes and plasmids, with all sequences and abundances known in advance.

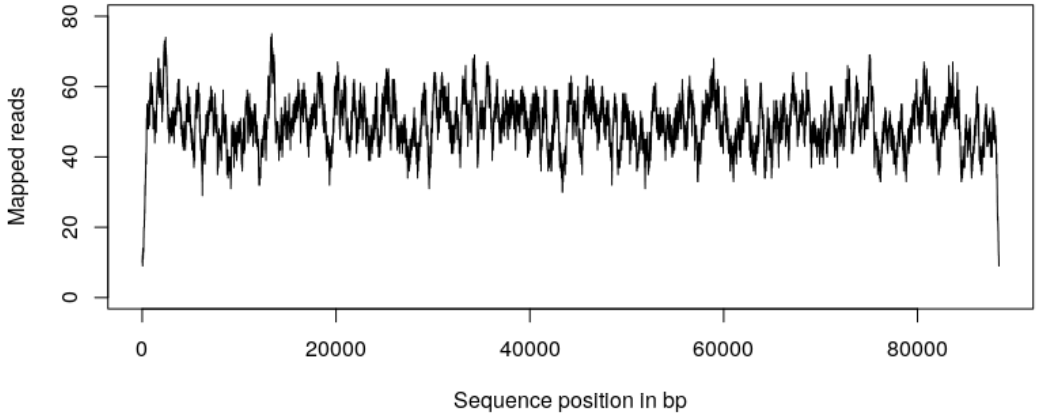
5.1.1 Creation of benchmark sequences

For the generation of simulated sequencing data, no sequencing simulator was available that would treat reference sequences as circular sequences. While simply concatenating the reference sequence to itself multiple times would bypass the problem, this needs a lot of memory for a large dataset. Thus, ART sequencing simulator[18] is extended to include reads that span the termini of the reference sequence. This is achieved by allowing fragments to be positioned at every position of the reference sequence, and extending the reference sequence in case the reads are mapping over the ends. A comparison of the normal ART sequencing simulator sampling and the circular sampling can be seen in figure 3.

Three simulated metagenenomic sequencing samples are generated, with sequencing parameters chosen to mimic a common short read metagenomic WGS experiment. The three samples represent varying levels of complexity and thus contain a community comprised of 20, 50 and 200 different prokaryotic species respectively. These species are selected randomly from the NCBI Refseq database, with the rule that their assemblies need to contain at least one plasmid. Further, the species sets is curated after selection, to contain at least one member of all major bacterial and archaeal clades present in the initial database, to counter the over-representation of model organisms and human pathogens. All species



(a) Circular mode



(b) Linear (default) mode

Figure 3: Example showing the effects of linear and circular sampling sequence simulation. 3a: Circular sampling using the modified ART sequence simulator. 3b: Linear sampling as is the default mode of the ART sequence simulator. Both simulations were done on the *E. coli* plasmid pESBL-EA11 with the parameters: art_illumina -ss MSv3 -p -sam -i /Path_to/NC_018659.1.fasta -rs 180506 -l 250 -f 50 -m 500 -s 25

within a sample community are then assigned a relative abundance and similarly, all plasmids are assigned a plasmid copy number.

All simulated sequencing samples consist of 5 million read pairs of length 150 bp, with a Illumina MSv1 error profile and a fragment length distribution with mean

Sample	Assembler	Total length	Contigs	Average length	N50
bm20	SPAdes	$7.27 \cdot 10^7$	32,108	2,263.44	10,152
bm20	MetaSPAdes	$7.35 \cdot 10^7$	40,635	1,809.57	6,967
bm50	SPAdes	$1.59 \cdot 10^8$	$1.2 \cdot 10^5$	1,333.21	3,003
bm50	MetaSPAdes	$1.62 \cdot 10^8$	$1.44 \cdot 10^5$	1,128.6	2,414
bm200	SPAdes	$3.06 \cdot 10^8$	$6.56 \cdot 10^5$	466.93	483
bm200	MetaSPAdes	$3.41 \cdot 10^8$	$8.7 \cdot 10^5$	391.64	355

Table 1: Summary of assembly statistics of the samples used for the benchmark

500 bp and a standard deviation of 50 bp. The three benchmark samples are then assembled using SPAdes and MetaSPAdes. The statistics of the assemblies can be seen in table 1. It becomes clear that the complexity between the three benchmark samples differs greatly, while the total length of the assemblies spans a factor of 4 in magnitude, the number of contigs increases roughly to a 20-fold from 20 to 200 species. The average length and the N50 drop substantially as well with increasing species numbers and shallower abundance slopes, indicating that the three samples really capture not just different sizes of metagenomic communities, but also reflect different levels of complexity in the assembly graph.

5.2 Initial evaluation of plasmid prediction methods

Using these three simulated samples, it is possible to look into the current problems and boundaries of existing plasmid prediction tools. The starting point for the improvement of plasmid prediction needs to be an assessment of the current boundaries of plasmid prediction. As the focus of this thesis is the prediction of individual plasmids, in this first analysis only tools that are capable of such predictions are investigated. These are MetaPlasmidSPAdes, which takes the raw reads as input and generates its own assemblies, as well as Recycler and

Score	MOB-Suite	Recycler	MetaPlasmidSPADes
Precision	0.63	0.41	0.69
Recall	0.55	0.24	0.28
F1	0.59	0.29	0.38

Table 2: Summary of plasmid prediction performance. Specifically, the performance of binning plasmidic sequences into their individual bins is evaluated. Thus, a bin is a true positive, if it contains 80% of a plasmids sequence and at least 80% of bins sequence corresponds to said plasmid. The scores presented are the average over all three benchmark samples (20, 50 and 200 species) assembled with MetaSPAdes (Recycler and MOB-Suite).

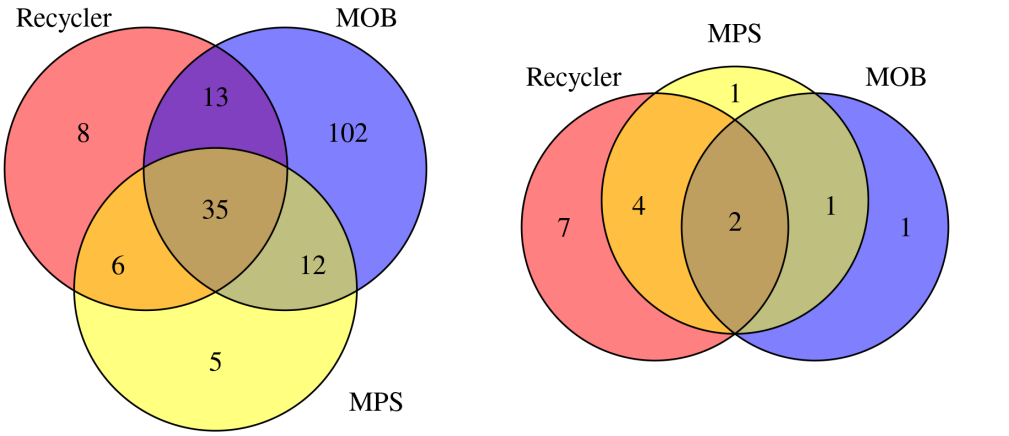
MOB-Suite, which take the finished assemblies as input and are run on both SPAdes and MetaSPAdes assemblies individually.

The predictive performance of the three programs is then evaluated by comparing the predicted plasmids to the references used in the sample creation. To analyse the performance on the goal of reconstructing individual, complete plasmids, every prediction is only considered to be a correct prediction if its sequences covers at least 80% of a single reference plasmid. To exclude bins containing multiple plasmids, more than 80% of the predicted sequence has to map to said plasmid. Table 2 shows the precision, recall and the F1 score the three programs achieved on average over all three samples, assembled by MetaSPAdes. This first analysis shows that sensitivity is overall low, with less than 0.3 in Recycler and MetaPlasmidSPAdes and slightly more than half of the references being recovered by MOB-Suite. Also, 30% to nearly 60% of bins are false positives, which can be either incomplete plasmids, chromosomal sequences or chimeras containing sequences of several references.

While the number of correctly predicted plasmids is comparatively high using the MOB-Suite, it also uses extensive a priori knowledge of plasmids, compared to the other three, purely assembly graph based methods. To assess whether

the approach of MOB-Suite is useful for the goal of this thesis, finding complete, novel plasmids, the prediction performance of MOB-Suite needs to be evaluated in the light of novel plasmids. This is done by excluding all plasmids that are represented in the MOB-Suite plasmid database with at least 10% sequence similarity and comparing the performance of the four plasmid prediction programs on this subset. As can be seen in Figure 4, the correct plasmid predictions of MOB-Suite are drastically reduced compared to Recycler and MetaPlasmidSPAdes when confronted with unknown plasmids. Thus, its approach is less suitable for the scope of this thesis, than the assembly graph based approaches of MetaPlasmidSPAdes and Recycler.

Contrary to the initial hypothesis of this thesis, Recycler finished its calculations faster than MOB-Suite, never exceeding the 20 minute mark for any sample.



(a) All plasmids

(b) Plasmids not in MOB-Suite database

Figure 4: Venn diagrams showing the correctly predicted plasmids of the 200 species sample, assembled by MetaSPAdes. The labels MPS and MOB are used for MetaPlasmidSPAdes and MOB-Suite respectively. Figure 4a shows the the predictions of all plasmids in the sample, while figure 4b excludes all plasmids that have a sequence similarity of more than 10% with a member of the MOB-Suite database

5.2.1 Erroneous plasmid prediction

Several features of plasmids in the assembly graph can be identified, that lead to a classification in terms of difficulties for their reconstruction.
The first category is the simple cycle, which is a plasmid represented in a single circular contig in the assembly graph. This is very easy to identify as a plasmid, due to being circular, complete and in most cases isolated.
A different category is the complex cycle, which is a plasmid, comprised of several contigs, but forming a cricle. The fragmentation is be due to internal repeats, which leads to more than a single circular path through the plasmids assembly graph. In other words, the problems arise in the plasmid not having an unambigous circular path, but rather several subcycles, which can lead to the plasmid being split up or not detected at all.
A similar category is the chimera, here the plasmid is comprised of several contigs, but has at least one shared sequence with another plasmid or a chromosome. The challenge in reconstructing plasmids of this category lies in finding a feature to identify the contigs of a single plasmid and tell them apart from the residual contigs. Such a feature could be coverage, GC content or sequence similarity to a known reference. If such an identification is not available or incomplete, the chimeric component can not be resolved, or the prediction will likely be erroneous.
The last and most hopeless category is the disjoint plasmid. This includes plasmids that are covered only partly due to low coverage, but also cases of assembly errors, where the plasmids contigs are not connected or do not form a cycle. In this case the plasmid can only be recovered in fragments, and only by the use of a priori contig features.

5.3 CAPP

This section will describe the ideas behind the development of a novel plasmid prediction program, as well as giving detailed information on the inner workings and performance of its individual parts.

5.3.1 Improvement upon current methods

The main concept of the novel program is, to build upon the existing methods of Recycler and MetaPlasmidSPAdes and to combine the main ideas of these algorithms to increase the sensitivity of plasmid prediction. The major concern of Recycler is the features used in determining plasmid like cycles. These features require plasmids to have equal coverage over their contigs in the assembly graph. However, this lacks a differentiation between high and low coverage distributions and it neglects the problem of repetitive or shared sequences, which have increased coverage. Also, the further filtering of classification of putative plasmids is done by merely making sure that paired end information of the reads matches the selected cycle, which allows for a high number of false positives.
The MetaPlasmidSPAdes method of selecting unambiguous cycles while applying a coverage cutoff, is great at selecting circular sequences that belong to a single molecular unit. However, the coverage cutoff is applied as a hard cap and globally, disregarding whether neighbouring contigs would fit the same molecule. Additionally, any plasmid containing an internal repeat or sharing a sequence with a higher coverage molecule will not be found as they can not produce unambiguous cycles.
Thus, a program utilizing plasmid like features and relative coverage cutoff for identifying candidate cycles, should be able to achieve higher sensitivity in plasmid reconstruction. Further, allowing for cyclic components that are likely to belong to a single plasmid to be presented as a solution similar to draft genomes

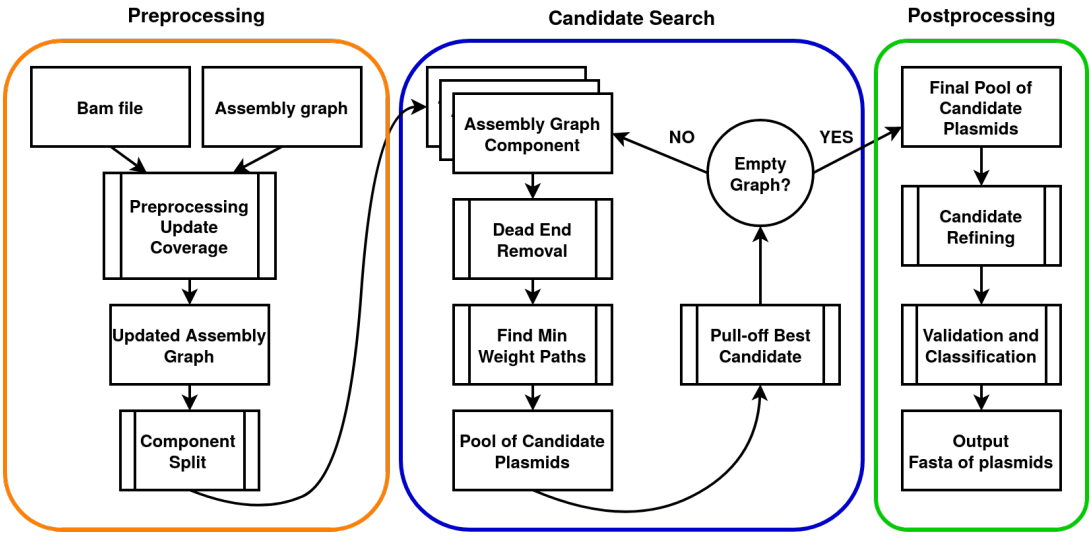


Figure 5: Flowchart of CAPP. The Process is divided into three segments, preprocessing, candidate search and postprocessing, as indicated by the orange, blue and green borders respectively. Data and objects are represented by simple rectangles, functions are shown as rectangles with double border. The circle in candidate search represents a decision step.

in genome assembly should open the possibility for plasmids with internal repeats to be predicted.

5.3.2 Overview

Using the identified problems and the presented ideas for improvement, a novel program is developed. It resembles the base of Recycler, which was initially thought to be improved in performance in the scope of this master thesis, however, a new implementation is needed, as Recycler is written in Python 2, which is no longer supported as of 1st of january 2020. As the code of Recycler is relatively simple, but only the basic concepts will be retained, the novel program is implemented from scratch in Python 3, with Recycler as conceptual blueprint. As the aim of the novel program is to combine aspects of Recycler and MetaPlasmidSPAdes, it is called a Combined Approach for Plasmid Prediction, or

CAPP. As can be seen in the flowchart of CAPP (figure 5), the program needs an assembly graph and an indexed bam file of reads mapped to the graph as input and is divided into three parts: Preprocessing, candidate search and post-processing. In Preprocessing, the program uses the read mapping to refine the assembly graph, as well as gathering the paired-end information and checking for reads spanning the edges of the graph. The updated assembly graph is then split into components to be individually processed.

Candidate search is the next segment of CAPP, here each component is analyzed for plasmid like cycles in the assembly graph. In the first step dead ends are pruned from the graph, as they are never part of a cycle and can therefore be excluded from the search. Next, each edge in the remaining component is used as a starting point for finding a minimal weight, plasmid-like cycle it is part of. Of the resulting pool of candidate cycles the best scoring candidate is accepted and pulled off the component. The remaining graph is searched again using the same steps, until all nodes and edges are removed by either dead end removal or candidate pull-off.

The last segment is postprocessing, where the pool of all accepted candidates is the starting point. The aim of the postprocessing is to refine candidates, remove false positives and provide additional information on the predictions. The refining step is carried out by merging candidates overlapping in the assembly graph, if they seem likely to belong to a single origin. After merging, the reamaining candidates are scored and validated using plasmid-like features and candidates that seem unlikely to be a complete and correct plasmid are filtered out. Concluding the program, the sequences of all final predictions are extracted from the assembly graph, concatenating the contigs according to the predicted path, and written to a multifasta file.

5.3.3 Utilization of read mapping

The first step in the preprocessing of CAPP is the preparation of the assembly graph. This includes reading the graph from the fastg file into memory and feeding it into a custom representation to facilitate adding additional information and options for manipulation in the later stages of CAPP. When reading the assembly graph, it is made sure that all nodes and edges are present on both strands of the assembly graph, as non viral DNA molecules are expected to be double stranded and to enable bidirectional traversal of the contigs. Next, the indexed bam file is used to correct the assembly graph as well as retrieve additional information, not represented in the graph. This means recalculating the coverage of the nodes, counting the number of reads spanning an edge to validate an edge beyond sequence overlap, but also the ends of read pairs are mapped to the graph and thus connections and spacing of nodes is evaluated on another level.

Since the contigs of an assembly graph consist of overlaps of length k at both ends of a node, with a unique sequence in between, coverage can only be assigned to the unique part of the node. Each read is checked for the proportion that is mapped to this unique sequence of the read, and the number of observed bases increased accordingly. If a read is terminal, the overlapping sequence is extracted and aligned to all nodes connected at this end. The alignment is a simple hamming distance and the bases of this overlap are assigned to the node with the least distance. Further, if such a terminal read spans the k-mer overlap and has at least one base aligned to the unique part of the node, the counter of observed reads for the edge connecting the two nodes is incremented by one. For all these operations the nodes and edges are strand specific. As the complementary strands of a chromosome or plasmid can be considered as a single unit when it comes to abundance in a sample, as well as the sequence and

orientation of sub-parts like contigs are, the coverage of a contig and its edges can be seen as the sum of both strands.
In addition to coverage and read pair information, the kmer overlap used in the graph is extracted and thus does not have to be manually specified as in Recycler. Further, the mean read length is calculated from a subset of reads (default 1000 reads) and similarly, the fragment length distribution is calculated from a set of read pairs that fully map the same node (default 10000).

5.3.4 Search for potential plasmids

Both Recycler and MetaPlasmidSPAdes use the idea of predicting plasmids in the assembly graph based on their circularity. While MetaPlasmidSPAdes is looking for unambiguous cyclocontigs based on coverage separation, Recycler uses scores to find a circular path with minimal weight for each edge. The latter idea will be used as the core prediction method in CAPP, as it seems possible to improve its sensitivity.
The algorithm for finding putative plasmids in the assembly graph is depicted in algorithm 1. This section will explain the details of this algorithm, with the exception of the minimal weight path search algorithm, the weight calculation and the solution pull-off, which are presented individually in chapters 5.3.5 to 5.3.7.
The algotithm operates on a representation of the assembly graph, which is reduced and altered along the way. To avoid loss of information, a copy of the original graph is used. Further, as cycles can only be found within a connected component, the graph is first split up into individual components which are analyzed sequentially.
The search itself is an iterative approach of finding candidates and reducing the graph by solution pull-off until the graph is empty or no more new solutions are

Algorithm 1: Search for potential plasmids in the assembly graph. Utilizes minimal weight path search (see algorithm 2).

```
   Data: G(V,E)
   Result: Set of approved candidate cycles
 1 candidates, accepted = new list ;
 2 while G is not empty do
 3 |   old candidates = candidates ;
 4 |   remove dead ends from G ;
 5 |   foreach edge e(u,v)∈E do
 6 |   |   weight, path ← FindMinWeightPath(e(u,v)) ;
 7 |   |   if path not empty then
 8 |   |   |   add (weight, path) to candidates ;
 9 |   |   end
10 |   end
11 |   if old candidates == candidates then
   |   |   // No new candidates - terminate search
12 |   |   break ;
13 |   end
14 |   candidates: remove duplicate paths, retain minimal weight ;
15 |   SortCandidates(candidates) ;
16 |   while candidates not empty do
17 |   |   weight, cycle ← pop(candidates) ;
18 |   |   if EvaluateCandidate(cycle) then
   |   |   |   // cycle passed quality check
19 |   |   |   add (weight, cycle) to accepted ;
20 |   |   |   PullOff(G, cycle) ;          // Pull cycle off graph G
21 |   |   end
22 |   |   if break condition then
23 |   |   |   break ;   // Remaining candidates are not good enough
24 |   |   end
25 |   end
26 end
27 return accepted
```

found. At the beginning of each iteration dead ends are pruned from the graph, as they are never part of any cycle. Next, a set of candidates is sampled by searching for the minimal weight path $p(v, u)$ for all edges $e(u, v)$, as explained

in section 5.3.5. If no new candidates are found in this step, the algorithm terminates.

Otherwise, any duplicates in the set of candidates are removed, with the lower weight being retained. Next, the candidates are sorted in a manner that correct predictions are likely to be first. This has been tried with three approaches, sorting by average weighted node coverage in descending order, as done in Recycler, by the total weight or the weight averaged by the number of nodes in the cycle, in ascending order respectively.

The sorted candidates are then evaluated individually and added to the accepted candidates if the quality criteria are met. While Recycler uses evaluation criteria as paired end read mapping and variation of coverage, the only quality criterium in CAPP is the minimum length of 1kb, as evaluation of candidates is performed in a later stage. This accepting of candidates continues until a break condition is met. This can be that a candidate is overlapping a recent accepted one or that the quality check was not passed.

Once the graph is empty, or no new candidates are found, all accepted candidates are returned and added to the pool of final candidates of all components, which are then evaluated as described in chapter 5.3.9.

5.3.5 Minimal weight path search

While the core idea of minimal weight cyclic paths being evaluated for all edges is retained, the implementation in CAPP is different to Recycler in many ways. The most impactful difference, the weight calculation, is explained in the next section of this thesis. The steps in candidate search can be split into: initial weighted cycle search, candidate sorting, candidate filtering, pull-off. The minimal weight cycle detection in Recycler is implemented by first calculating and setting the weights of all edges based on the current assembly graph, then us-

ing Dijkstras algorithm to find the shortest weighted path $p(v, u)$ for every edge $e(u, v)$ in the graph. In contrast, CAPP needs to calculate the weights during path search, as they are context specific to the starting edge $e(u, v)$. Further, CAPP uses a modified version of Dijkstras algorithm, described in algorithm 2, to allow for revisiting of nodes, which can resolve internal repeats when using weights which are context specific to the current candidate path.

Algorithm 2: Minimal Weight cycle detection. A modified version of Dijkstras shortest path algorithm, allowing for revisiting nodes.

Data: Assembly graph G(V,E) including node and edge parameters length and coverage
Input: Edge e(u,v) to be evaluated, maximum distance md (threshold parameter)
Result: Minimal weight path p(v,u), if any exists

```
1  Paths, Scores = new dictionary;
2  foreach successor s of v do
3  |  p = list(s) ;                                          // p:path
4  |  t = False ;                                  // t:termination flag
5  |  d = 0 ;                          // d:distance since t was set
6  |  Paths ← GetWeight(v,s): s, p, d, t ;
7  end
8  while Paths not empty do
9  |  w, q, p, d, t ← pop(min(Paths)) ;      // w:weight, q:last node
10 |  if q not in Scores then
11 |  |  Scores ← q: w ;                     // path to q is minimal
12 |  |  remove t if set ;
13 |  |  if q == u then
14 |  |  |  return w, p ;                              // Cycle found
15 |  |  end
16 |  else
17 |  |  if t then
18 |  |  |  d += len(q) ;
19 |  |  |  if d > md then
20 |  |  |  |  terminate path
21 |  |  |  end
22 |  |  else
23 |  |  |  t = True, d = 0 ;          // Flag path for termination
24 |  |  end
25 |  end
26 |  foreach successor s of q do
27 |  |  Paths ← GetWeight(q,s): s, p+s, d, t ;
28 |  end
29 end
30 w = -1, p = empty list ;                          // No cycle found
31 return w, p
```

5.3.6 Weight calculation

The weights in Recycler are based on the concept that plasmids will have a higher coverage than chromosomal contigs and tries to favor trustworthy nodes. Equation 7 shows how the weights for each edge are created in Recycler. The product of the length and coverage of the starting node of the edge are used to reduce the applied weight the greater they are.

$$e(v,u) \in E; \quad v,u \in V \tag{6}$$

$$s_{\mathrm{e(v,u)}} = \frac{1}{cov_{\mathrm{v}} * len_{\mathrm{v}}} \tag{7}$$

While being a valid approach, plasmids do not always have higher coverage than genomic sequences, even more so in a metagenomic setting. Further, the length of a contig does not correlate with its origin, with the exception of very large contigs, which are more likely to belong to a chromosome, as most plasmids are magnitudes smaller than the typical bacterial chromosome.

With these considerations, the weights implemented in CAPP try to be more considerate on promoting paths representing complete plasmids. Along with the circularity, uniform coverage is a feature expected to be found in plasmids. As presented before, repeats and shared sequences break this concept. The impact of this problem can be mitigated in two ways, the edges in an assembly graph are less prone to being shared or repeatedly used than nodes, and these phenomena lead to coverage peaks, but not coverage drops, which leaves the lower boundary

of the predicted uniform coverage intact.

$$e(a,b),\ e(v,u) \in E; \quad a,b,v,u \in V \tag{8}$$

$$X = Pois(\lambda = reads_{\text{target}}) \tag{9}$$

$$reads_{\text{target}} = reads_{\text{e(a,b)}} \tag{10}$$

$$s_{\text{edge}} = -\log_2(\ Pr(X > reads_{\text{target}} + |reads_{\text{target}} - reads_{\text{e(v,u)}}|)\ ) \tag{11}$$

$$rf_{\text{e(v,u)}} = \frac{1}{cov_{\text{v}} * len_{\text{v}}} \tag{12}$$

$$s_{\text{e(v,u)}} = s_{\text{edge}} * rf_{\text{e(v,u)}} \tag{13}$$

Equations 8 - 13 show the weight calculation of CAPP. Conceptually, the starting edge $e(a,b)$ is used as a primer for a putative plasmid. The coverage of this putative plasmid is set to the coverage estimated by the starting edge, and the expected number of reads spanning an edge belonging to said plasmid, $reads_{\text{target}}$, is set to the number of reads observed on the starting edge. To evaluate the fit of the coverage of an edge to the postulated plasmid, the distribution of read counts of edges is modeled as Poisson distribution. The edge score s_{edge} is then calculated as the negative binary logarithm of the probability inferred from the Poisson distributed random variable and the observed number of reads of the current edge $e(v,u)$. In order to favor high coverage and long nodes, an additional factor equal to the weight as computed by Recycler is used. The final weight is then the product of the edge score and the Recycler factor.

5.3.7 Solution pull-off

The concept of solution pull-off, as presented by Rozov et al., is to simplify the assembly graph by subtracting accepted solutions and therefore uncovering plasmids with lower coverage. Nodes and edges with coverage of 0 are then

pruned from the network. This strategy promises to help resolve overlapping plasmids, especially if one of the overlapping plasmids can be identified while the others are hidden.

CAPP uses this method, just as Recycler, however the subtraction of the plasmid works differently. Recycler uses the average node coverage of the cycle to be pulled off and then subtracts it from all nodes that are part of the solution. However, as repeats can lead to higher coverage and CAPP deals with edge coverages as well, the median node and edge coverage are subtracted from the nodes and edges of the solution respectively. This tries to mitigate the impact of high coverage nodes or edges and reduce the bias introduced into the assembly graph.

5.3.8 Performance improvements

In order to run the program on larger datasets, it needs to be ensured that the complexity is reduced as much as possible. Several improvements are implemented to improve performance. One of the improvements targets the reduction of the search space when finding the minimal weight cycles in a component. If the candidate is selected by the criterium of the least highest single edge weight, when choosing only one cycle per iteration, the path exploration can be stopped for every path which has a single edge weight higher than the least highest edge weight of the current iteration. This leads to a reduction of the search space and thus a runtime improvement.

Another improvement targets the recalculation of path weights. To reduce recalculation of path weights, every step that is taken in path exploration is stored in memory. To reduce memory consumption, all paths are stored as trees. The root is the starting node, every internal node is a node in the assembly graph and traversal of the tree from root to leaf thus encodes a path in the assembly

graph. In addition to the node IDs, the weight of the path is stored as well. Thus, in all iterations after the initial, this path tree is traversed to restore the whole search space from memory, without recalculating any weights. When the best solution of a minimal weight cycle search is pulled off the assembly graph, viable solutions still remain. However, as some cycles are overlapping each other in the assembly graph, the solutions of the last iteration might have changed and thus are not possible or optimal anymore. Therefore, all IDs of nodes and edges that are modified during solution pull-off or dead-end removal are stored. When a modified node is encountered during tree traversal, the subtree containing the modified node is removed and the path up to this point is added in its current state to the list of paths to search. After the search space is loaded from the tree, the minimal weight cycle search is resumed, again starting with the current lowest scoring path. This ensures that paths which now might have a lower score than the previous best solution are now processed and found before the previous best solutions. In the case that no scores have changed in the search space of an edge, the previous solution is accepted again at once, as it still has the lowest total score.

A different approach to speed up the minimal weight cycle search, is to pull off multiple candidates during the same iteration. To make sure that the pull-off of one candidate does not change the score of another one, the solution are processed sequentially, all the modified nodes are stored and the acception of candidates continues until all candidates are accepted, or the next candidate cycle includes a node that has been modified. That way the number of iterations can be reduced. It needs to be noted that this strategy conflicts with the search space limiting strategy to provide a premature stop criterium for all paths which would not beat the current top scoring path.

As some edges are not part of any cycle, without any restrictions, the path

search will search a path to every node, until it will state that no cycle has been found. This is computationally expensive and should be avoided, especially in large components. Therefore an option is provided in CAPP to set a limit to number of iterations when searching for the minimal weight path at any given edge.

5.3.9 Candidate evaluation and classification

The initial minimal weight cycle search accepts all candidates that form a cycle until the set of accepted candidates subtracted from the assembly graph does not contain any more acceptable cycles. This leads to many candidates being of chromosomal origin, or otherwise incorrect plasmidic cycles (chimeras or subcycles). To increase the predictive power of the program, two different actions are performed on the candidate cycles: improvement of the initial candidates and filtering out likely false positives.

The improvement of the initial candidates covers the incomplete subcycles of plasmids. In the case that a plasmid has an internal repeat, the plasmid sequence can be split into at least two subcycles and givenin a table: that they are shorter than the whole plasmid cycle, they will most likely be preferred in a minimal weight cycle search. However, overlapping candidates can be merged, to create the complete plasmid if all its subcycles are found and selected. The task gets more complex in the light of repeats being shared between different plasmids or chromosomes. Thus, overlapping cycles have to be analysed for compatibility. In this program, the criteria for merging is a compatibility of the coverage of the edges of the overlapping candidates, as it is expected that different DNA molecules in a metagenomic sample will often have a different abundance and thus a different coverage.

One criterium for merging two candidate cycles is their coverage similarity. More

specifically, the median edge read count is used as proxy, again using a Poisson distribution to model impact of the difference in coverage (see equation 16).

$$c, d \in candidate\ cycles \tag{14}$$

$$X = Pois(\lambda = median(reads_{\mathrm{c}}) \tag{15}$$

$$merge\ if : 0.2 < Pr(X < median(reads_{\mathrm{d}}) \tag{16}$$

A different criterium uses paired end information over triplets of nodes. If two nodes share read pairs, but are only connected in the graph via a third node, then this path is supported by paired end information. Conversely, if no read pairs are shared between two such nodes, but both have supported paths to different nodes over said third middle node, then paired end information disagrees with said path. This concept is useful in cycles with repeats, that were split into subcycles. These subcycles are then favored by minimal weight path search, as they are shorter, but, given sufficient sequencing depth, they show paths unsupported by paired end information. Such candidate cycles can then be merged if the number of unsupported node triplets is reduced by merging.

Filtering out incorrect candidates is the other task in processing the candidates. Here, all possible classes of incorrect predictions need to be looked into: chromosomal cycles, plasmid subcycles, chimeric candidates comprised of both only plasmids and a combination of plasmidic and chromosomal sequences. Here, several criteria are used to classify the candidates. The first criterium is uniform coverage, more specifically drops in coverage are investigated. As the sequencing process can be regarded as mostly random in the sampling of reads, despite some known biases, the coverage of edges and nodes in a candidate cycle can be expected to follow a normal or poisson distribution. This distribution is violated if nodes and edges of different DNA molecules are mixed. The reason that

only coverage drops are investigated, is that repeats, regardless whether they are internal or external, lead to a peak in coverage, which would also violate the assumption of a uniform coverage distribution. To counter this problem, the lowest node and edge coverage are tested for the probability to find a node or edge with their respective coverage or less, under the assumption that the coverage distribution is a poisson distribution with lambda equal to the median of the candidates node or edge coverage. Again, the median is used as an approximation for lambda instead of the mean to counter biases by coverage peaks introduced by repeats or shared sequences.
While the filtering is a necessary and still unsolved part of plasmid prediction, the current implementations in CAPP do not provide consistent improvement of specificity or the F1 score, as can be seen in the comparison of the filtered and unfiltered output of CAPP in figure 6.

5.3.10 Additional features

In addtion to the core functionality of CAPP, some additional features have been included. One of them is the option to output the revised assembly graph, or only the assembly graph. Therefore, the preprocessing is available for users, regardless of their interest in plasmid prediction.

5.4 Benchmark of plasmid prediction methods

In this thesis, the already existing tools PlasmidSPAdes, MetaPlasmidSPAdes, MOB_Suite, Recycler, PlasFlow, PlaScope and PlasClass are benchmarked together with the newly developed CAPP. This includes both the quality of the prediction or classification as well as the runtime an memory requirements. The benchmark uses three different simulated sequencing samples of different levels of complexity, includes both chromosomes and plasmids and treats these molecules

as circular in the simulated sequencing process.

5.4.1 Performance comparison - precision and sensitivity

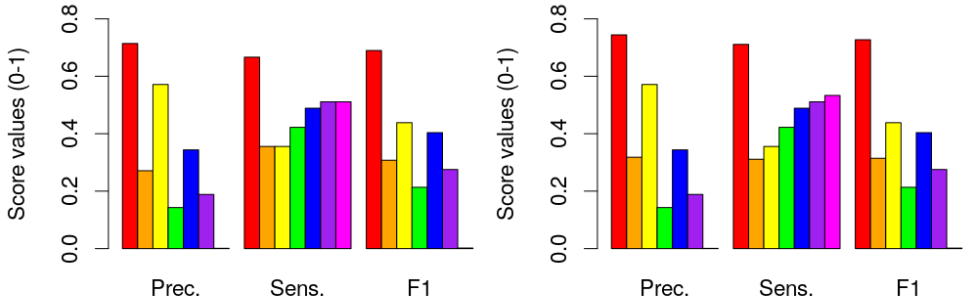
The prediction performance is evaluated with two different aims: the binning of individual plasmids, as is the goal of Recycler, MOB-Suite, PlasmidSPAdes, MetaPlasmidSPAdes and CAPP, and the separation of contigs classifying them as origininating from chromosome or plasmid. In both cases confusion matrix scores are calculated, as described in section 4.1.7.

The prediction of binning individual plasmids is shown in figure 6. The figure shows the overall trend, that increasing complexity reduces the prediction performance, as can be observed in the F1 score of the different methods. Further, MOB-Suite is consistently high performing compared to all other programs, with the only exception of MetaPlasmidSPAdes, which is more specific (but less sensistive) in the 50 and 200 species samples. However, MOB-Suite uses databases for prediction, and, as shown in figure 4, performs worse than Recycler and MetaPlasmidSPAdes on novel plasmids.

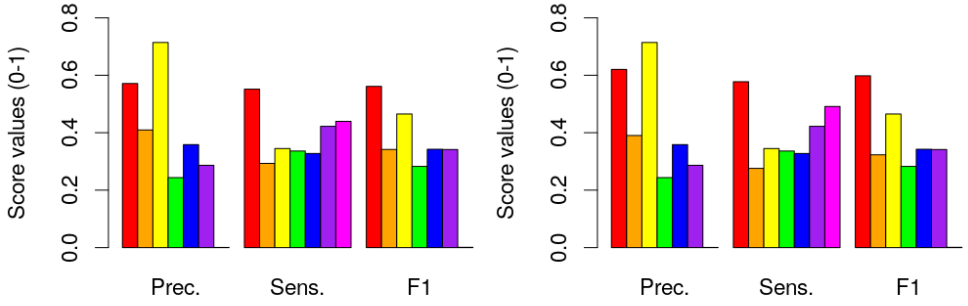
When looking at the purely assembly graph based methods Recycler, PlasmidSPAdes, MetaPlasmidSPAdes and CAPP, MetaPlasmidSPAdes consistently has the best F1 score. While its precision is high, in most cases its sensitivity is below the other three methods.

The performance of CAPP is evaluated in a complete and an unfiltered mode, without filtering out candidates after merging. This is done, as the evaluation and filtering of candidates is not beneficial in all cases, as can be seen in the F1 scores of the 50 and 200 species samples. The unfiltered version of CAPP shows improved sensitivity in finding complete cycles based only on the assembly graph, compared to Recycler, PlasmidSPAdes and MetaPlasmidSPAdes.

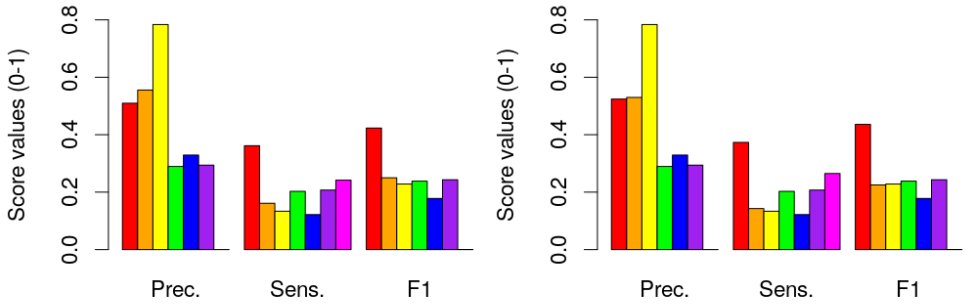
As a comparison, or baseline of predictive power, the contigs of the assembly



(a) 20 species



(b) 50 species



(c) 200 species

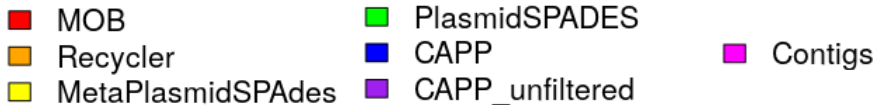


Figure 6: Comparison of the plasmid prediction performance of individual plasmid bins. The figures 6a to 6c show the performance on the benchmark samples with 20, 50 and 200 species respectively. The plot on the left shows results for SPAdes assembly, on the right are results for MetaSPAdes assembly. The colours of denote the different programs tested as seen in the legend. Prec.: Precision, Sens.: Sensitivity

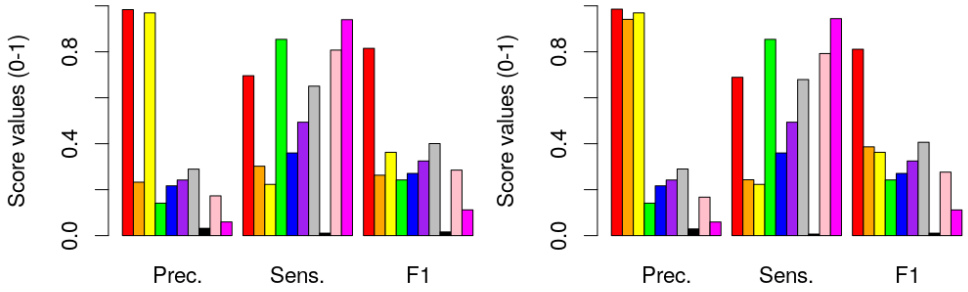
were included in the evaluation, with each contig being treated as individual bin and as if it were a predicted plasmid. It should be noted, that MOB-Suite is the only program to achieve higher sensitivity in binning individual plasmids than the contigs achieve themselves. The predictive power in binning is however close to 0, as the precision is very poor, due to the huge number of contigs and no included filtering. A different aim in plasmid prediction is the segregation of contigs by their origin: chromosome or plasmid. While only PlaScope, Plasflow and PlasClass were developed specifically for that goal, the predictions of all programs can be evaluated in the light of this aim. The results of this evaluation on the three benchmark samples can be seen in figure 7. As this is a binary classification, a simple perfomrnace baseline can be set by classifying all contigs as plasmidic.

The evaluation shows, that PlaScope has highly variable precision, but very low sensitivity. Throughout all samples and assembly methods, PlaScope never surpasses the F1 score of the baseline.

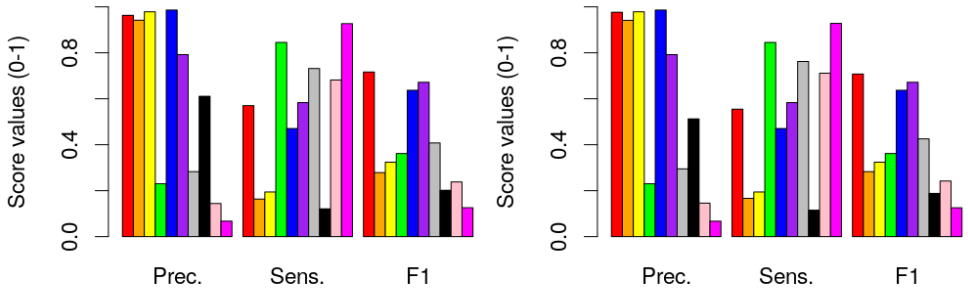
Plasflow and PlasClass show consistent performance in all samples, with plasflow having higher precision and F1 scores in all samples, but less sensitivity in the 20 and 200 species samples, which is the exact opposite behavior as reported by Pellow et al.

The performance of MOB-Suite shows high precision and sensitivity in this task as well, its F1 score is higher than any of the methods developed for this task, with the only exception being Plasflow in the 200 species sample.

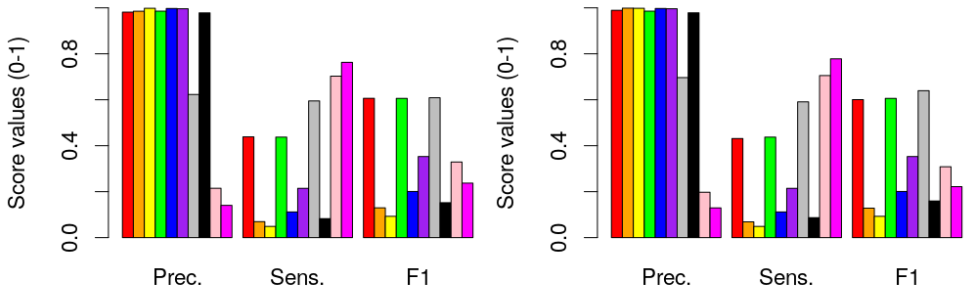
Recycler, CAPP and MetaPlasmidSPAdes show high precision in the 50 and 200 species samples, indicating that, while precision in binning individual plasmids is variable, the predicted sequences mostly originate from plasmids. However, these methods show low sensitivity compared to Plasflow and PlasClase, especially with increasing complexity. Their predictive power is highly variable and



(a) 20 species



(b) 50 species



(c) 200 species

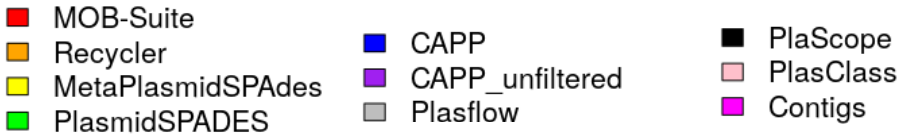
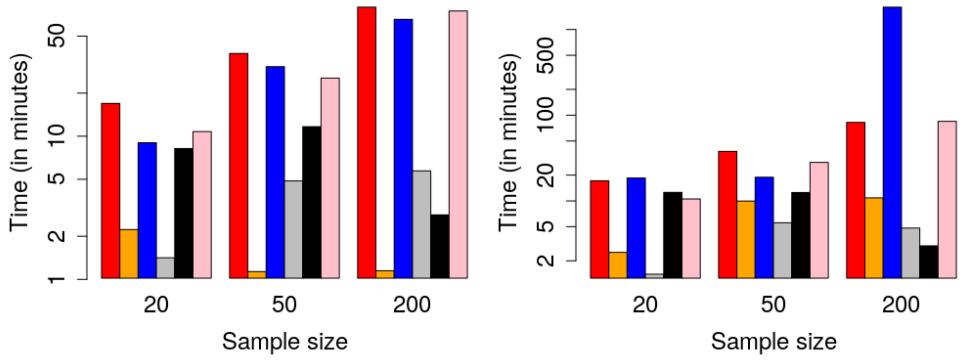


Figure 7: Evaluation of plasmid and chromosome sequence segregation. The figures 7a to 7c show the performance on the benchmark samples with 20, 50 and 200 species respectively. The left and right plot show results of SPAdes and metaSPAdes assemblies respectively. The colours of denote the different programs tested as seen in the legend. Prec.: Precision, Sens.: Sensitivity

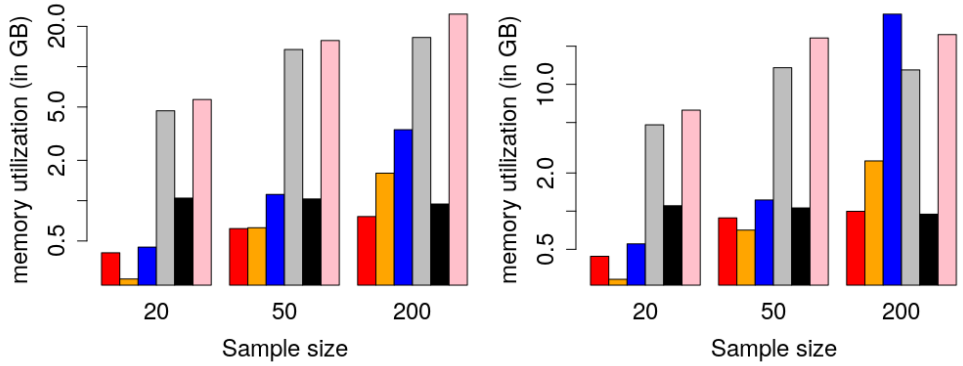
in most cases the F1 score is below that of Plasflow.
What can be seen in this evaluation is, that the choice of the assembler hardly had any impact the predictive on the power of the methods,with the exception of Recycler in the 20 species sample.

5.4.2 Performance comparison - runtime and memory

In addition to the predictive power, the runtime and memory utilization were analyzed for all plasmid prediction tools. Excluded from this evaluation are however MetaPlasmidSPAdes and PlasmidSPAdes, as they both included their own assembly of the sequencing samples and thus had to run on a different compute node due to higher memory and runtime needs. All other programs are compared without any preprocessing steps as assembly and read mapping. The runtime and memory utilization on the 20, 50 and 200 species samples, including both SPAdes and MetaSPAdes assembly, can be seen in figure 8. It should be noted, that the machine learning methods, Plasflow, PlaScope and PlasClass, are multithreaded and were allowed to use up to 8 threads.
The choice of the assembler has a visible impact on both the runtime and the memory utilization of Recycler and CAPP. These are also the only programs that work with directly with the assembly graph, which is more complex in the MetaSPAdes assembly (see table 1).
The runtime evaluation shows, that CAPP is on a similar level to MOB-Suite, and uses less memory than Plasflow and PlasClass in all but the 200 species MetaSPAdes assembled sample. However, it uses more memory and needs more compute time than Recycler in all samples.



(a) Runtime comparison (left: SPAdes, right: MetaSPAdes)



(b) Memory utilization comparison (left: SPAdes, right: MetaSPAdes)

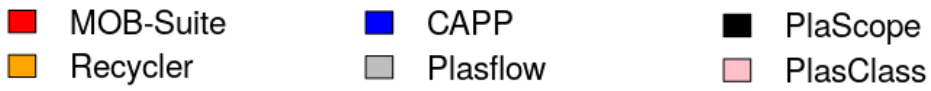


Figure 8: Comparison of runtime and memory consumption of plasmid prediction programs. The left and right plots show results of SPAdes and metaSPAdes assemblies respectively. The colours of denote the different programs tested as seen in the legend.

6 Discussion

This thesis presents benchmark of state of the art plasmid prediction methods using three different simulated sequencing samples. While all of these tools, including Recycler, managed to finish their calculations on even the most complex sample within 2 hours, this evaluation reveals many shortcomings in the predic-

tive power of the different approaches and tools. CAPP, a novel tool for plasmid prediction is developed for this thesis, utilizing and improving upon existing approaches of Recycler and MetaPlasmidSPAdes.
In this section the state of the art in plasmid prediction in metagenomic sequencing samples will be discussed as well as the individual strengths and weaknesses of the evaluated tools. Further, future developments of both the field of plasmid prediction and CAPP will be considered.

6.1 State of the art

The plasmid reconstruction benchmark in this thesis shows the performance of all sate of the art tools for the detection of plasmid sequences in metagenomic sequencing samples. This evaluation reveals that the current approaches have limited capability in perfoming this task, as seen in the low F1 scores. Apart from the general predictive power, this benchmark also identifies strengths and weaknesses of the individual methods.
If one is interested in known plasmids, rather than the detection of novel ones, MOB-Suite is the best performing tool in this benchmark. However, it is unclear how recombination and mutation of plasmid sequences affects the performance of MOB-Suite, as in this benchmark the reference sequences were directly taken from ncbi RefSeq and not altered in any way. Further, a naive database search using a recent plasmid database and established methods as blast, may yield the same, or better results in this setting.
The recently published PlasClass was developed to improve upon PlasFlow.[35] In this benchmark however, PlasFlow reached higher F1-scores in all settings. This may be due to the scores in this thesis being weighted by the length of the contigs, while Pellow et al. used the contig numbers instead. For the task of correctly retrieving plasmid sequences rather than a classification of contigs,

PlasFlow yields better results in the scope of this benchmark.
In the setting of the aim of this thesis, predicting complete plasmids without bias against novel sequences, MetaPlasmidSPAdes has the highest F1 score in 2 out of 3 samples. It has precision that could not be reached by Recycler and CAPP and was surpassed by MOB-Suite in only the least complex sample. The drawback of this tool is its dependence on coverage difference, leading to low sensitivity in complex communities with a shallow abundance distribution.
While Recycler is able to predict more plasmids correctly in the complex scenario, this is not the case in any other sample. CAPP is the most sensitive method for reconstructing complete and individual plasmids, when run in the unfiltered mode, outperforming both Recycler and MetaPlasmidSPAdes. This more extensive list of putative plasmids can help researchers find and confirm more novel plasmids in downstream experiments than would be possible using any other method.
SPAdes and MetaSPAdes output the assembly graph including nodes and edges, but scaffolded contigs are output as well. In this thesis, these were used as a baseline for predictive power in plasmid prediction and curiously, the contigs contained more plasmids that can be considered complete and correctly binned as defined in the evaluation, than Recycler, MetaPlasmidSPAdes, PlasmidSPAdes or CAPP throughout all samples and assemblies. Even though it is unclear how many of these plasmids can be cirularized, this might be a promising ressource to include into plasmid prediction programs working directly on the assembly graph.

6.2 Future of the field

As the field of metagenomics grows and sequencing technologies advance in throughput and cost, the need for tools able to reconstruct plasmids of com-

plex metagenomic samples will increase. This thesis reveals that current tools are not at a satisfactory state in accomplishing this task. While CAPP shows that an increase in sensitivity is possible, the impact on predictive power is inconsistent and lacks in specificity compared to other methods.

With the availability of long read sequencing technologies as PacBio and Oxford NanoPore, the sequencing of complete plasmids in a single read becomes possible.[45, 46] The assembly graph in general can be expected to become less ambiguous, with the effect of large plasmids being easier to disentangle from the remaining chromosomal sequences. While these technologies are promising, they bring their own challenges with them, as increased cost and higher error rates. Thus, predicting plasmids in samples sequenced with these technologies will need different approaches and tools than with short read sequencing. If the problem of distinguishing plasmids from chromosomes will however be solved by the long reads alone, is hard to predict.[14]

A different crossroads in plasmid prediction is the use of artificial intelligence in contig classification or graph based methods. While the classification specificity of the graph based methods was shown to be higher for the majority of methods and samples, the sensitivity was higher for machine learning approaches. With rapid improvements in the field of artificial intelligence and new ways to utilize DNA sequence information, it seems possible, that this class of methods will reach better classification performance than graph based algorithms.

With more work on plasmids in metagenomic samples, increasing amounts of plasmid sequences in databases can be expected. This will benefit methods like MOB-Suite, that extract known plasmids, but also improve training data for machine learning tools. However, it is unclear if this will benefit the detection of unknown plasmids, or help reconstruct plasmids with major recombinations or mutations compared to plasmid sequences in databases.

In spite of the three simulated sequencing data samples being less complex and messy than real data, the benchmark in this thesis showed major shortcomings of state of the art plasmid prediction tools. Thus, a critical assessment using samples of real sequencing experiments, and more complex simulated samples is needed, to fully be aware of the current blind spot of these methods.
With more precise and improved protocols for plasmid extraction from metagenomic samples, assessment of the performance of these state of the art plasmid prediction tools on plasmidome samples will be necessary, as most of these tools were implemented specifically for sequencing samples that include chromosomal sequences.

6.3 Outlook

While CAPP is in a complete and functioning state as it is used in the benchmark, there are several additions to be implemented and future improvements that can be thought of. To ensure CAPP can be used by a broad audience of biologists, some limitations need to be lifted and usability needs to be improved. CAPP currently needs the assembly graph in the format of fastg and only works with the SPAdes representation of overlaps. The adaptation of the fastg reader and possibly adding a parser for the gfa format will be beneficial.
While there are already some strategies in place to reduce runtime and memory utilization, further improvements have to be made to allow for more complex data sets. For example, multi-threading can be expected to be implemented without complications, as the search for minimal weight paths is independent for each edge, reducing the compute time of the most demanding step.
CAPP was developed specifically for paired end short read data. As only parts of the scoring and classification are dependent on read pair information, it is possible ot adapt CAPP to work with single end data, as well as long reads,

however, the impact on the predictive power is unclear.

In addition to these improvements, the filtering and evaluation of the predicted plasmids needs to be improved, as the current implementation does not always benefit the overall quality of the prediction. This is, however, an unsolved problem in plasmid prediction and further testing and evaluation is needed. For example, the current implementation of CAPP, using features of the assembly graph to classify the candidate plasmids, could be refined for consistent removal false candidates. Another approach was taken by Antipov et al., who implemented a machine learning classifier for the predicted plasmids of MetaPlasmidSPAdes, called PlasmidVerify[4], which could be tested on the output of CAPP or other tools as well.

An additional feature of CAPP that is planned to be implemented, is providing confidence on the predicted plasmids. Plasmids which are represented as simple cycles can be marked as high confidence. Further, confidence scores and classification as potential chimera or partial plasmid can be included, but will require more work. However, with the low F1 scores of plasmid prediction methods in general, guidance on decisions what predictions are trustworthy will benefit planning of downstream experiments.

6.4 Assumptions and limitations

Within this thesis and CAPP, several assumptions were made, which lead to limitations in applicability of the results and tools. First of all, the definition of plasmids is limited to circular DNA molecules, while linear plasmids and chromids have been proven to exist as well.

Further, the simulated samples used in the benchmark have reduced complexity. The simulated microbial communities and their structures ignore strain variation and DNA amplification bias of dividing cells. In addition, the simulated

sequencing process does not inlcude the GC bias in amplification[9], contamination or degradation of sample DNA as well as any form of human error. As the resulting sequencing data contains less variation and errors as can be expected of real samples, no filtering and trimming steps were included.

6.5 Conclusion

In this thesis, the state of the art plasmid prediction tools were evaluated and benchmarked using simulated metagenomic sequencing data. The sequencing simulation treated the plasmids as circular molecules, which is crucial for prediction methods relying on circularity. Further, the synthetic communities varied in complexity and were modeled to have realistic abundance structures.
Evaluation of available methods revealed low sensitivity in detection of completely novel plasmids. A new program for plasmid prediction, CAPP, was developed, combining methods and approaches of Recycler and MetaPlasmidSPAdes. This novel tool is able to reconstruct more plasmids correctly than Recycler or MetaPlasmidSPAdes.

Supplemental material

- Table S1: Detailed information on simulated sequencing samples
- Table S2: Detailed information on the evaluation of plasmid prediction
- CAPP: Source code of CAPP package
- Scripts: Source code of the scripts used to generate sample communities and simulate the sequencing process
- ART: Source code of the modified ART sequencing simulator

Figures

Tables

References

[1] Rustam I. Aminov. Horizontal gene exchange in environmental microbiota. *Frontiers in Microbiology*, 2(JULY):1–19, 2011.

[2] Mizue Anda, Yoshiyuki Ohtsubo, Takashi Okubo, Masayuki Sugawara, Yuji Nagata, Masataka Tsuda, Kiwamu Minamisawa, and Hisayuki Mitsui. Bacterial clade with the ribosomal RNA operon on a small plasmid rather than the chromosome. *Proceedings of the National Academy of Sciences of the United States of America*, 112(46):14343–14347, 2015.

[3] Dmitry Antipov, Nolan Hartwick, Max Shen, Mikhail Raiko, Alla Lapidus, and Pavel A. Pevzner. PlasmidSPAdes: Assembling plasmids from whole genome sequencing data. *Bioinformatics*, 32(22):3380–3387, 2016.

[4] Dmitry Antipov, Mikhail Raiko, Alla Lapidus, and Pavel A. Pevzner. Plasmid detection and assembly in genomic and metagenomic data sets. *Genome Research*, 29(6):961–968, 2019.

[5] Sergio Arredondo-Alonso, Rob J. Willems, Willem van Schaik, and Anita C. Schürch. On the (im)possibility of reconstructing plasmids from whole-genome short-read sequencing data. *Microbial Genomics*, 3(10), 2017.

[6] Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A. Gurevich, Mikhail Dvorkin, Alexander S. Kulikov, Valery M. Lesin, Sergey I. Nikolenko, Son Pham, Andrey D. Prjibelski, Alexey V. Pyshkin, Alexander V. Sirotkin, Nikolay Vyahhi, Glenn Tesler, Max A. Alekseyev, and Pavel A. Pevzner. SPAdes: A new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 19(5):455–477, 2012.

[7] Christopher W. Beitel, Lutz Froenicke, Jenna M. Lang, Ian F. Korf, Richard W. Michelmore, Jonathan A. Eisen, and Aaron E. Darling. Strain- and plasmid-level deconvolution of a synthetic metagenome by sequencing proximity ligation products. *PeerJ*, 2014(1):1–19, 2014.

[8] Alessandra Carattoli, Ea Zankari, Aurora Garciá-Fernández, Mette Voldby Larsen, Ole Lund, Laura Villa, Frank Mløler Aarestrup, and Henrik Hasman. In Silico detection and typing of plasmids using plasmidfinder and plasmid multilocus sequence typing. *Antimicrobial Agents and Chemotherapy*, 58(7):3895–3903, 2014.

[9] Yen Chun Chen, Tsunglin Liu, Chun Hui Yu, Tzen Yuh Chiang, and Chi Chuan Hwang. Effects of GC Bias in Next-Generation-Sequencing Data on De Novo Genome Assembly. *PLoS ONE*, 8(4), 2013.

[10] L. C. Cook and G. M. Dunny. Effects of biofilm growth on plasmid copy number and expression of antibiotic resistance genes in Enterococcus faecalis. *Antimicrobial Agents and Chemotherapy*, 57(4):1850–1856, 2013.

[11] J. W. Decousser, G. Royer, C. Branger, M. Dubois, D. Vallenet, C. Médigue, and E. Denamur. PlaScope: a targeted approach to assess the plasmidome from genome assemblies at the species level. *Microbial Genomics*, 4(9):1–8, 2018.

[12] Julián R. Dib, Martin Wagenknecht, María E. Farías, and Friedhelm Meinhardt. Strategies and approaches in plasmidome studies-uncovering plasmid diversity disregarding of linear elements? *Frontiers in Microbiology*, 6(MAY):1–12, 2015.

[13] Merly Escalona, Sara Rocha, and David Posada. A comparison of tools for the simulation of genomic next-generation sequencing data. *Nature Reviews Genetics*, 17(8):459–469, 2016.

[14] Sophie George, Louise Pankhurst, Alasdair Hubbard, Antonia Votintseva, Nicole Stoesser, Anna E. Sheppard, Amy Mathers, Rachel Norris, Indre Navickaite, Chloe Eaton, Zamin Iqbal, Derrick W. Crook, and Hang T.T. Phan. Resolving plasmid structures in enterobacteriaceae using the MinION nanopore sequencer: Assessment of MinION and MinION/illumina hybrid data assembly approaches. *Microbial Genomics*, 3(8):1–8, 2017.

[15] Sachin Kumar Gupta, Shahbaz Raza, and Tatsuya Unno. Comparison of denovo assembly tools for plasmid metagenome analysis. *Genes and Genomics*, 41(9):1077–1083, 2019.

[16] Jo Handelsman, Michelle R. Rondon, Sean F. Brady, Jon Clardy, and Robert M. Goodman. Molecular biological access to the chemistry of unknown soil microbes: A new frontier for natural products. *Chemistry and Biology*, 5(10), 1998.

[17] Peter W. Harrison, Ryan P.J. Lower, Nayoung K.D. Kim, and J. Peter W. Young. Introducing the bacterial 'chromid': Not a chromosome, not a plasmid. *Trends in Microbiology*, 18(4):141–148, 2010.

[18] Weichun Huang, Leping Li, Jason R. Myers, and Gabor T. Marth. ART: A next-generation sequencing read simulator. *Bioinformatics*, 28(4):593–594, 2012.

[19] Judith Ilhan, Anne Kupczok, Christian Woehle, Tanita Wein, Nils F. Hülter, Philip Rosenstiel, Giddy Landan, Itzhak Mizrahi, and Tal Dagan. Segregational Drift and the Interplay between Plasmid Copy Number and Evolvability. *Molecular Biology and Evolution*, 36(3):472–486, 2019.

[20] Michael Jahn, Carsten Vorpahl, Thomas Hübschmann, Hauke Harms, and Susann Müller. Copy number variability of expression plasmids determined by cell sorting and droplet digital PCR. *Microbial Cell Factories*, 15(1):1–12, 2016.

[21] Daehwan Kim, Li Song, Florian P Breitwieser, and Steven L Salzberg. Centrifuge: rapid and accurate classificaton of metagenomic sequences, version 1.0.4_beta. *bioRxiv*, 26(12):054965, 2016.

[22] Ankita Kothari, Yu Wei Wu, John Marc Chandonia, Marimikel Charrier, Lara Rajeev, Andrea M. Rocha, Dominique C. Joyner, Terry C. Hazen, Steven W. Singer, and Aindrila Mukhopadhyay. Large circular plasmids from groundwater plasmidomes span multiple incompatibility groups and are enriched in multimetal resistance genes. *mBio*, 10(1):1–15, 2019.

[23] Pawel S. Krawczyk, Leszek Lipinski, and Andrzej Dziembowski. PlasFlow: predicting plasmid sequences in metagenomic data using genome signatures. *Nucleic acids research*, 46(6):e35, 2018.

[24] Cedric C. Laczny, Valentina Galata, Achim Plum, Andreas E. Posch, and Andreas Keller. Assessing the heterogeneity of in silico plasmid predictions based on whole-genome-sequenced clinical isolates. *Briefings in Bioinformatics*, 20(3):857–865, 2017.

[25] Val F. Lanza, María de Toro, M. Pilar Garcillán-Barcia, Azucena Mora, Jorge Blanco, Teresa M. Coque, and Fernando de la Cruz. Plasmid Flux in Escherichia coli ST131 Sublineages, Analyzed by Plasmid Constellation Network (PLACNET), a New Method for Plasmid Reconstruction from Whole Genome Sequences. *PLoS Genetics*, 10(12), 2014.

[26] Heng Li and Richard Durbin. Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics*, 26(5):589–595, 2010.

[27] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, and Richard Durbin.

The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[28] Serghei Mangul, Lana S. Martin, Brian L. Hill, Angela Ka Mei Lam, Margaret G. Distler, Alex Zelikovsky, Eleazar Eskin, and Jonathan Flint. Systematic benchmarking of omics computational tools. *Nature Communications*, 10(1):1–11, 2019.

[29] Guillaume Marçais, Arthur L. Delcher, Adam M. Phillippy, Rachel Coston, Steven L. Salzberg, and Aleksey Zimin. MUMmer4: A fast and versatile genome alignment system. *PLoS Computational Biology*, 14(1):1–14, 2018.

[30] Niranjan Nagarajan and Mihai Pop. Sequence assembly demystified. *Nature Reviews Genetics*, 14(3):157–167, 2013.

[31] Stephen Nayfach, Zhou Jason Shi, Rekha Seshadri, Katherine S. Pollard, and Nikos C. Kyrpides. New insights from uncultivated genomes of the global human gut microbiome. *Nature*, 568(7753):505–510, 2019.

[32] Sergey Nurk, Dmitry Meleshko, Anton Korobeynikov, and Pavel A. Pevzner. MetaSPAdes: A new versatile metagenomic assembler. *Genome Research*, 27(5):824–834, 2017.

[33] Nathan D. Olson, Todd J. Treangen, Christopher M. Hill, Victoria Cepeda-Espinoza, Jay Ghurye, Sergey Koren, and Mihai Pop. Metagenomic assembly through the lens of validation: Recent advances in assessing and improving the quality of genomes assembled from metagenomes. *Briefings in Bioinformatics*, 20(4):1140–1150, 2018.

[34] Alex Orlek, Nicole Stoesser, Muna F. Anjum, Michel Doumith, Matthew J. Ellington, Tim Peto, Derrick Crook, Neil Woodford, A. Sarah Walker, Hang Phan, and Anna E. Sheppard. Plasmid classification in an era of whole-genome sequencing: Application in studies of antibiotic resistance epidemiology. *Frontiers in Microbiology*, 8(FEB):1–10, 2017.

[35] David Pellow, Itzik Mizrahi, and Ron Shamir. PlasClass improves plasmid sequence classification. *PLoS computational biology*, 16(4):e1007781, 2020.

[36] Jörn Petersen, John Vollmers, Victoria Ringel, Henner Brinkmann, Claire Ellebrandt-Sperling, Cathrin Spröer, Alexandra M. Howat, J. Colin Murrell, and Anne Kristin Kaster. A marine plasmid hitchhiking vast phylogenetic and geographic distances. *Proceedings of the National Academy of Sciences of the United States of America*, 116(41):20568–20573, 2019.

[37] Franziska Pfeiffer, Carsten Gröber, Michael Blank, Kristian Händler, Marc Beyer, Joachim L. Schultze, and Günter Mayer. Systematic evaluation of error rates and causes in short samples in next-generation sequencing. *Scientific Reports*, 8(1):1–14, 2018.

[38] Miguel Pignatelli and Andrés Moya. Evaluating the fidelity of De Novo short read metagenomic assembly using simulated data. *PLoS ONE*, 6(5), 2011.

[39] Andrey D. Prjibelski, Irina Vasilinetc, Anton Bankevich, Alexey Gurevich, Tatiana Krivosheeva, Sergey Nurk, Son Pham, Anton Korobeynikov, Alla Lapidus, and Pavel A. Pevzner. ExSPAnder: A universal repeat resolver for DNA fragment assembly. *Bioinformatics*, 30(12):293–301, 2014.

[40] James Robertson and John H.E. Nash. MOB-suite: software tools for clustering, reconstruction and typing of plasmids from draft assemblies. *Microbial genomics*, 4(8):1–7, 2018.

[41] Roye Rozov, Aya Brown Kav, David Bogumil, Naama Shterzer, Eran Halperin, Itzhak Mizrahi, and Ron Shamir. Recycler: An algorithm for detecting plasmids from de novo assembly graphs. *Bioinformatics*, 33(4):475–482, 2017.

[42] Alexander Sczyrba, Peter Hofmann, Peter Belmann, David Koslicki, Stefan Janssen, Johannes Dröge, Ivan Gregor, Stephan Majda, Jessika Fiedler, Eik Dahms, Andreas Bremges, Adrian Fritz, Ruben Garrido-Oter, Tue Sparholt Jørgensen, Nicole Shapiro, Philip D. Blood, Alexey Gurevich, Yang Bai, Dmitrij Turaev, Matthew Z. Demaere, Rayan Chikhi, Niranjan Nagarajan, Christopher Quince, Fernando Meyer, Monika BalvočiutÄ—, Lars Hestbjerg Hansen, Søren J. Sørensen, Burton K.H. Chia, Bertrand Denis, Jeff L. Froula, Zhong Wang, Robert Egan, Dongwan Don Kang, Jeffrey J. Cook, Charles Deltel, Michael Beckstette, Claire Lemaitre, Pierre Peterlongo, Guillaume Rizk, Dominique Lavenier, Yu Wei Wu, Steven W. Singer, Chirag Jain, Marc Strous, Heiner Klingenberg, Peter Meinicke, Michael D. Barton, Thomas Lingner, Hsin Hung Lin, Yu Chieh Liao, Genivaldo Gueiros Z. Silva, Daniel A. Cuevas, Robert A. Edwards, Surya Saha, Vitor C. Piro, Bernhard Y. Renard, Mihai Pop, Hans Peter Klenk, Markus Göker, Nikos C. Kyrpides, Tanja Woyke, Julia A. Vorholt, Paul Schulze-Lefert, Edward M. Rubin, Aaron E. Darling, Thomas Rattei, and Alice C. McHardy. Critical Assessment of Metagenome Interpretation - A benchmark of metagenomics software. *Nature Methods*, 14(11):1063–1071, 2017.

[43] Masaki Shintani, Zoe K. Sanchez, and Kazuhide Kimbara. Genomics of microbial plasmids: Classification and identification based on replication and transfer systems and host taxonomy. *Frontiers in Microbiology*, 6(MAR):1–16, 2015.

[44] Kornelia Smalla, Sven Jechalke, and Eva M. Top. Plasmid Detection, Characterization, and Ecology. *Plasmids*, 3(1):445–458, 2015.

[45] Yoshihiko Suzuki, Suguru Nishijima, Yoshikazu Furuta, Jun Yoshimura, Wataru Suda, Kenshiro Oshima, Masahira Hattori, and Shinichi Morishita. Long-read metagenomic exploration of extrachromosomal mobile genetic elements in the human gut. *Microbiome*, 7(1):1–16, 2019.

[46] Tonya L. Taylor, Jeremy D. Volkening, Eric DeJesus, Mustafa Simmons, Kiril M. Dimitrov, Glenn E. Tillman, David L. Suarez, and Claudio L. Afonso. Rapid, multiplexed, whole genome and plasmid sequencing of foodborne pathogens using long-read nanopore technology. *Scientific Reports*, 9(1):1–11, 2019.

[47] Sonia R. Vartoukian, Richard M. Palmer, and William G. Wade. Strategies for culture of 'unculturable' bacteria. *FEMS Microbiology Letters*, 309(1):1–7, 2010.

[48] John Vollmers, Sandra Wiegand, and Anne Kristin Kaster. *Comparing and evaluating metagenome assembly tools from a microbiologist's perspective - Not only size matters!*, volume 12. 2017.

[49] Haina Wang, Nan Peng, Shiraz A. Shah, Li Huang, and Qunxin She. Archaeal Extrachromosomal Genetic Elements. *Microbiology and Molecular Biology Reviews*, 79(1):117–152, 2015.

[50] Mukta M. Watve, Neelesh Dahanukar, and Milind G. Watve. Sociobiological control of plasmid copy number in bacteria. *PLoS ONE*, 5(2), 2010.

[51] Agata Wesolowska-Andersen, Martin I. Bahl, Vera Carvalho, Karsten Kristiansen, Thomas Sicheritz-Pontén, Ramneek Gupta, and Tine R. Licht. Choice of bacterial DNA extraction method from fecal material influences community structure as evaluated by metagenomic analysis. *Microbiome*, 2(1):1–11, 2014.

[52] Ryan R. Wick, Mark B. Schultz, Justin Zobel, and Kathryn E. Holt. Bandage: Interactive visualization of de novo genome assemblies. *Bioinformatics*, 31(20):3350–3352, 2015.

[53] Chunying Zhong, Donghai Peng, Weixing Ye, Lujun Chai, Junliang Qi, Ziniu Yu, Lifang Ruan, and Ming Sun. Determination of plasmid copy number reveals the total plasmid DNA amount is greater than the chromosomal DNA amount in Bacillus thuringiensis YBT-1520. *PLoS ONE*, 6(1), 2011.

[54] Fengfeng Zhou and Ying Xu. cBar: A computer program to distinguish plasmid-derived from chromosome-derived sequence fragments in metagenomics data. *Bioinformatics*, 26(16):2051–2052, 2010.