Date of acceptance Grade

Instructor

MetaFlow: Metagenomics taxonomic analysis using network flows

Ahmed Sobih

Helsinki May 26, 2016

M.Sc. Thesis

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Ahmed Sobih

Työn nimi — Arbetets titel — Title

MetaFlow: Metagenomics taxonomic analysis using network flows

Oppiaine — Läroämne — Subject
Computer Science - Bioinformatics

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
M.Sc. Thesis	May 26, 2016	38 pages + 6 appendices

Tiivistelmä — Referat — Abstract

Our planet is pervaded by hundreds of millions of microorganisms that are not visible to the naked eye. These microorganisms, also known as *microbes*, include bacteria, archaea, fungi, protists and viruses. *Metagenomics* allows for the study of microbial samples collected directly from the environment without prior culturing. A crucial step in metagenomics analysis is to unveil the structure of the microbial community in a specific environment; this step is called *metagenomics taxonomic analysis* (or *community profiling*).

In this thesis we explain what is metagenomics taxonomic analysis, why it is important, and we present MetaFlow, a new tool for solving the metagenomics community profiling problem using high-throughput sequencing data. MetaFlow estimates the richness and the abundances at species taxonomic rank, based on coverage analysis across entire genomes, and it is the first method to apply network flows to solve this problem. Experiments showed that MetaFlow is more sensitive and precise than popular tools such as MetaPhlAn and mOTU, and its abundance estimation is better by 2-4 times. MetaFlow is available at https://github.com/alexandrutomescu/.

ACM Computing Classification System (CCS):
Mathematics of computing ∼ Network flows
Applied computing ∼ Bioinformatics
Applied computing ∼ Computational genomics

Avainsanat — Nyckelord — Keywords
Metagenomics community profiling, flow networks, high-throughput sequencing, minimum-cost flow

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Acknowledgment

I would like to express my deep gratitude to my supervisor Prof. Veli Mäkinen for his support, patience, and guidance during my work on this master thesis. I would also like to extend my sincere thanks to my instructor Dr. Alexandru Tomescu for his guidance throughout the research process, and his instructive feedback while writing the thesis. Furthermore, I would like to thank the Computer Science Department at University of Helsinki for funding the research project that led to this thesis. I wish to thank Dr. Fabio Cunial and Dr. Antti Honkela for giving me the chance to start working on metagenomics analysis.

Finally, I would like to thank my wife and my family for their great support and encouragement throughout my study.

Contents

1 Introduction

1.1 From microscopes to metagenomics

Our planet is pervaded by hundreds of millions of microorganisms that are not visible to the naked eye. These microorganisms, also known as *microbes*, include bacteria, archaea, fungi, protists and viruses. They are ubiquitous in water, air, soil and in our bodies, and they play a vital role in maintaining the balance of our ecosystem. For example, half of the oxygen on Earth is produced by marine microbes [RLL+03], and the human microbiome contains at least 100 times as many genes as our own genome [GPD+06]. Therefore, studying these microorganisms and understanding their impact on our life is essential.

The hypothesis of the existence of microorganisms dates back to many centuries. However, the first actual observation of microbes was not before the invention of the microscope in the 17th century. Since then, scientists worked relentlessly in order to conquer and decipher this mysterious world.

The discovery of the DNA structure in 1953 revolutionized all life sciences. It answered one of the most difficult questions that haunted the scientists for a long time: how genetic instructions are passed on from one generation to the next. The answer fundamentally changed our understanding of genetics. It is the DNA that holds the genetic instructions, not the proteins as it was thought before. Together with the invention of DNA sequencing technology, they made a tremendous impact on microbiology. The microbiologists got a completely different view on the microscopic world. Previously, identification of the microbes was only based on their phenotypic characteristics (i.e., the observable characteristics such as morphology). Nowadays, they can be accurately identified based on their genotypic characteristics as well. For example, the 16S rRNA can be used for identifying bacteria, as it is highly conserved between different bacterial species [WBPL91].

Genomics, which is the study of the structure and the functions of individual genomes, has triggered a major advance in our understanding of microbes. The microbial genomes studies started with sequencing *Bacteriophage MS2* RNA in 1976 [FCD+76], and since that time a large number of microbial genomes were sequenced and studied. Despite their essential role in microbiology, genomics solely is not sufficient for understanding the microbial world, for important reasons. As a matter of fact, microbes do not exist in isolatation. On the contrary, they live in communities, and they interact with each others and with their host [SBT+13]. Such

kind of interactions cannot be studied using clonal culturing. Moreover, the number of microbial cells on Earth is estimated to be 10^{30} [TG08], and the majority of them cannot be cultured in labs due to limitations in the current technology [ALS95].
In 1985, it was proposed for the first time to clone the DNA collected directly from the environment [PSLO85]. This opened the doors for the emergence of a new branch in bioinformatics called *metagenomics* (the term was coined in 1998 [HRB+98]). Metagenomics allows for the study of microbial samples collected directly from the environment without prior culturing in labs. Metagenomics studies revealed the fact that there are numerous microbial organisms that we do not know about. Those microbial organisms could not be captured using clonal culturing. Thus, metagenomics proved to be a powerful tool that helps reveal the secrets of the microbial world.
The invention of high-throughput sequencing (HTS) and the daily improvements in sequencing techniques paved the way for huge advances in metagenomics. Nowadays, sequencing machines are capable of generating hundreds of millions of reads at ever-dropping cost, which makes sequencing thousands of genomes in metagenomics samples an easy task.
Metagenomics studies incorporates different types of analysis, such as taxonomic analysis (or community profiling), functional analysis, protein signature-based analysis, and comparative analysis [HMJ+12]. In this thesis, we focus on taxonomic analysis, which is a crucial step in metagenomic analysis process. We explain what is taxonomic analysis, why it is important, and we present MetaFlow, a new taxonomic analysis tool for solving the community profiling problem using high-throughput sequencing data.

1.2 Metagenomics taxonomic analysis

A crucial step in metagenomics analysis is to unveil the structure of the microbial community in a specific environment. This analysis provides valuable information which can help solve challenging problems in different domains like ecology [RLY+11], agriculture [HTAC+07], and personalized medicine and public health [NW10]. For example, changes in human microbiota have been linked to obesity [BDW+04] and Crohn's disease [MTS+12].
The goal of metagenomics taxonomic analysis is to estimate the species richness (i.e., the number of different microbial species present in a given sample), and the species abundances (i.e., their relative frequencies). Figure 1 shows the metagenomics tax-

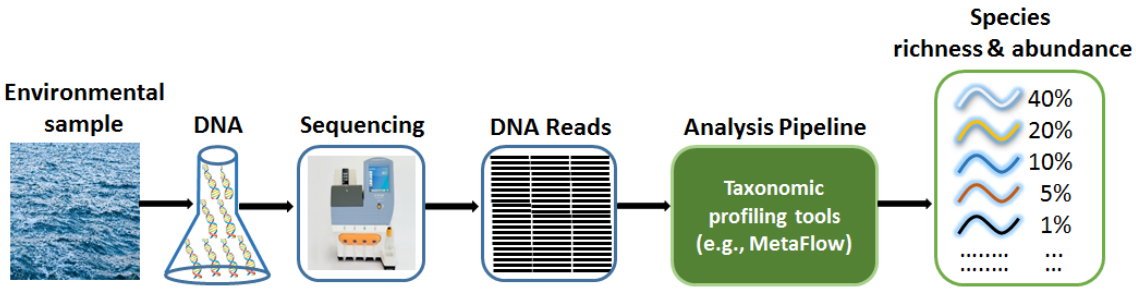


Figure 1: Metagenomics taxonomic profiling steps.

onomic profiling steps. For many reasons, such as the genotypic similarity between different species (i.e., similarity between some regions in their genomes), sequencing biases and errors, and the incompleteness of the microbial genomes databases, metagenomics taxonomic analysis is a challenging task.

2 Preliminaries

In this chapter, we explain the preliminary concepts required for understanding our work. Our definitions of some biology-related concepts (e.g., sequence alignment) are tailored to metagenomics for the purpose of understanding our work, and there are broader definitions for those concepts.

2.1 DNA sequencing and sequence alignment

DNA sequencing is the process of determining the correct order of the four nucleotides bases (thymine, adenine, cytosine and guanine) within a DNA molecule (e.g., a bacterial or a human genome). The length of bacterial genomes ranges from approximately 130 thousand base-pairs (bps) to 14 million bps [HLP+13], and the human genome contains approximately 3 billion bps [VAM+01]. Due to limitations in the current technology, DNA sequencing machines are incapable of generating the sequence of a complete genome in one go. Instead, they generate short fragments of DNA sequences called *reads*. Depending on the sequencing technology used, the read length ranges from tens of bps to thousands of bps.

In metagenomics, the generated DNA reads are coming from multiple microbial genomes and are mixed together. One method for identifying from which microbes these DNA reads are coming from is to match the reads against a database of mi-

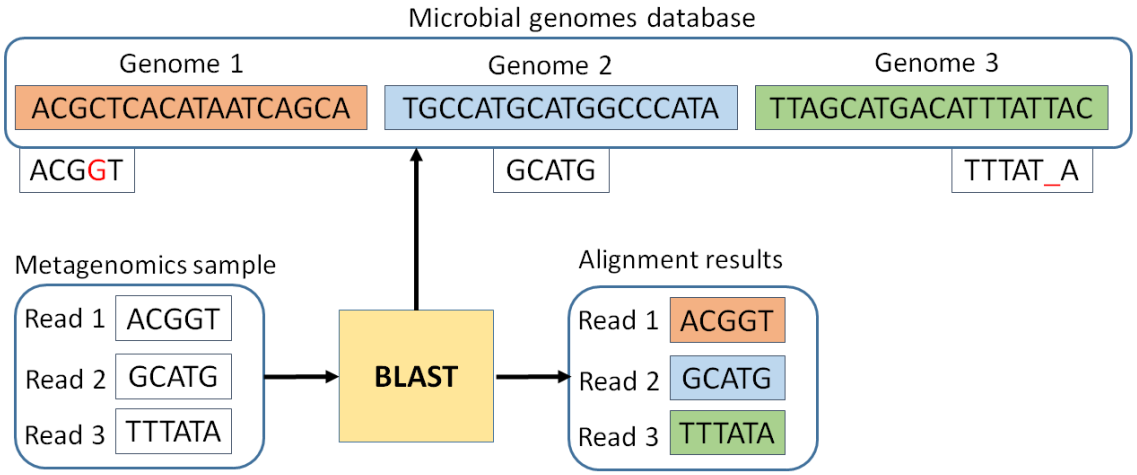


Figure 2: Sequence alignment for a metagenomic sample. Genome 1, 2, and 3 represent a microbial genomes database. Read 1, 2, and 3 represent the DNA reads in our metagenomic sample. BLAST will map read 1 to genome 1 since this is the best match. It will map read 2 and 3 to genome 2 and 3, respectively.

crobial genomes. This process is called *sequence alignment*. There are many tools for sequence alignment, e.g., BLAST [AGM+90]. BLAST can be used to align the metagenomics reads, and to obtain a match between each read and one of the genomes in the database (as shown in Figure 2). However, this method is not sufficient for estimating the richness and the species abundances, since any read can be perfectly aligned to multiple genomes due to the genotypic similarity between different species. In addition, the bacterial genomes databases are incomplete. As a consequence, many reads will not be mapped to any genome. Furthermore, even if we have a single perfect match, it does not mean it is the correct one since the read might be originating from another species that does not exist in the database. As we shall explain later, there are different approaches for solving these problems.

2.2 Taxonomy classification

Scientists classify biological organisms into groups based on some shared characteristics and assign names to those groups. Similar organisms (e.g., *L.acidophilus* & *L.agilis*) are grouped together into a *taxon* (e.g., *Lactobacillus*) and are given a *taxonomic rank* (e.g., Genus); similar taxa at a given rank are further aggregated to create a super taxon at a higher rank, thus forming a taxonomic hierarchy. One way for defining the similarity between organisms is based on the similarity between

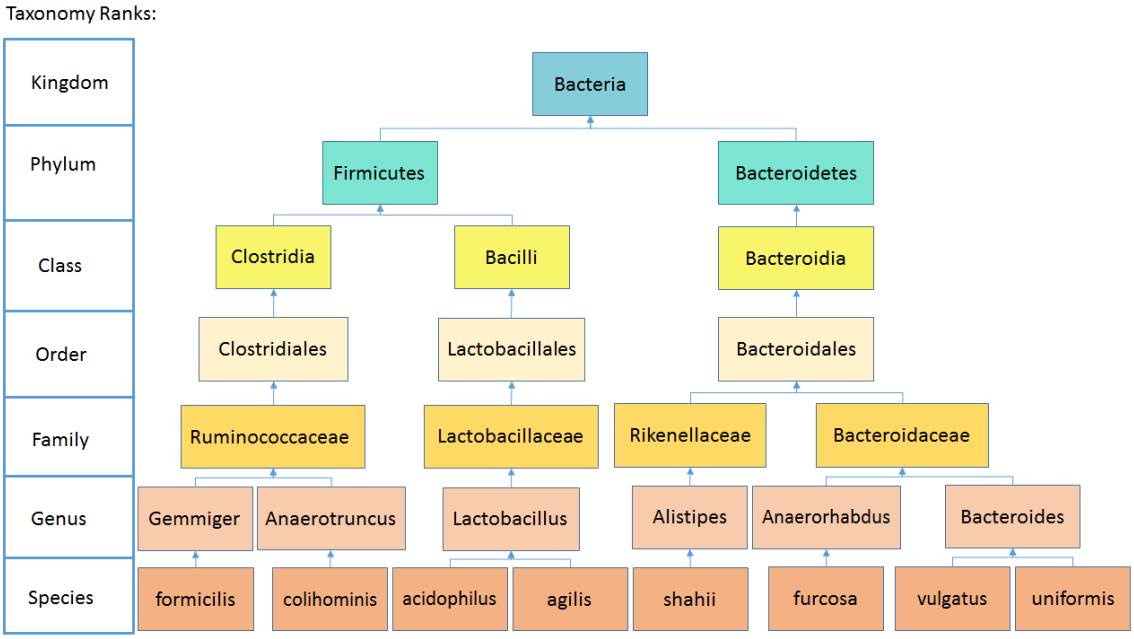


Figure 3: An example for the taxonomy tree of some bacterial species. There are seven taxonomic ranks (Species, Genus, Family, Order, Class, Phylum, and Kingdom). The bacterial species at species-rank are grouped together based on their similarity to form a new taxon at genus-rank (e.g., species *B.uniformis* and *B.vulgatus* are grouped together in the *Bacteroides* genus). Similar genera are grouped together to form a new taxon at family-rank, and so forth.

their genetic characteristics. For example, species with very similar DNA will be grouped together to form a taxon at genus-rank. Figure 3 shows an example for a taxonomy tree.

In metagenomics, classifying the read at the species taxonomic rank is preferred over classifying it at a higher taxonomic rank like genera or family. For example, specifying that a DNA read is originating from (Kingdom: *Bacteria*, Phylum: *Firmicutes*, Class: *Bacilli*, Order: *Lactobacillales*, Family: *Lactobacillaceae*, Genus: *Lactobacillus*, Species: *acidophilus*) is better than (Kingdom: *Bacteria*, Phylum: *Firmicutes*, Class: *Bacilli*, Order: *Lactobacillales*, Family: *Lactobacillaceae*), because it provides more information for the microbiologists. There is a trade-off between the specificity and the accuracy of prediction, and some tools choose lower specificity (i.e., classification at a high taxonomic rank) if they cannot accurately estimate the abundances at species-rank.

2.3 Graphs and flow networks

Our solution for the metagenomic profiling problem is based on *flow networks*. We model the problem as an optimization problem, specifically as *a minimum-cost flow problem* on a *bipartite graph*. In an *optimization problem* it is required to find a solution which minimizes (maximizes) a certain cost (utility) among all possible solutions of the problem. A *flow network* is *a directed graph* where each edge has a capacity, demand and cost, and receives a flow. It is required to find an optimal way of sending some content through this network, such that the amount of flow on an edge does not exceed the capacity of the edge. Flow networks can be used for solving a wide variety of optimization problems, and they have been used to solve many problems in bioinformatics. For example, in [TKRM13] flow networks were used to estimate transcript expression, and in [WAC+08] a network flow based method was used to assemble the HCV Quasispecies. Our tool is the first to apply flow networks to solve the metagenomic profiling problem. Below we give the definitions necessary for understanding our work. The definitions 2.3.3 - 2.3.5 are from [MBCT15].

Definition 2.3.1.
A graph*, is a tuple* $G = (V, E)$ *where where* V *is a set of* vertices *and* $E \subseteq \{\{u, v\} : u, v \in V\}$ *is a set of* edges*.*

A graph is called a *directed graph* if the edges connecting the vertices are directed from one vertex to another. If the graph has weights (e.g., integers) associated to its edges, we call it a *weighted graph*. A directed graph without *cycles* (i.e., a walk, determined by the direction of the edges, which starts and ends at the same vertex) is called a *directed acyclic graph*.

Definition 2.3.2. Bipartite graph
A bipartite graph is a graph $G = (V, E)$ *in which the vertex set* V *can be divided into two disjoint subsets* A, B *such that every edge* $e \in E$ *has one end in* A*, and the other end in* B*.*

Definition 2.3.3. Flow network
A flow network is a tuple $N = (G, l, u, c, q)$*, where*

- $G = (V, E)$ *is a directed graph with a unique source* $s \in V(G)$ *that has only outgoing flow, and a unique sink* $t \in V(G)$ *that has only incoming flow;*
- l *and* u *are functions assigning a non-negative demand and capacity, respectively, to every arc;*

- *c is a function assigning a cost per unit of flow to every arc;*
- *q is the required value of the flow.*

A flow over a flow network is a function satisfying the flow conservation property, the demand and capacity constraints, and having a given value.

Definition 2.3.4. Flow over a flow network
A flow over a flow network $N = (G, l, u, c, q)$ *is a function* $f\colon E(G) \to Q_+$ *that fulfills the following conditions:*

- *Flow conservation: for every vertex* $x \in V(G) \setminus \{s, t\}$*, it holds that:*

$$\sum_{y \in N^-(x)} f(x, y) = \sum_{y \in N^+(x)} f(x, y).$$

- *Demand and capacity constraints: for every arc* $(x, y) \in E(G)$*, it holds that:*

$$l(x, y) \leq f(x, y) \leq u(x, y).$$

- *Required flow value q:*

$$\sum_{y \in N^+(s)} f(s, y) = \sum_{y \in N^-(t)} f(x, t).$$

Definition 2.3.5. Minimum cost flow problem
Given a flow network $N = (G, , u, c, q)$*, find a flow* f *over* N *that minimizes:*

$$cost(f) = \sum_{(x,y) \in E(G)} c(x, y) f(x, y).$$

3 Related work

The selection of the approach and the tools for metagenomics taxonomic analysis depends on the laboratory method used for collecting and sequencing the sample, and the final goal of the analysis. There are two main methods for sequencing the metagenomic samples: the shotgun sequencing and the 16S rRNA sequencing. Our tool is designed for analyzing the shotgun sequencing samples, thus, after briefly explaining the difference between the two sequencing methods, we will focus on the analysis approaches for shotgun sequencing.

3.1 Shotgun sequencing *versus* 16S rRNA sequencing

In shotgun sequencing, the whole DNA materials in the sample are sequenced. On the other hand, in 16S rRNA sequencing, only some phylogenetic marker genes, which are conserved genes that can classify species at a specific taxonomic rank, e.g., 16S rRNA genes, are extracted and sequenced. While shotgun sequencing generates tens of millions of reads, 16S rRNA sequencing generates a much lower number of reads, which is cheaper and faster to analyse. However, 16S rRNA sequencing analysis does not provide accurate estimations at species-rank. Experiments showed that the analysis results for 16S-rRNA metagenomic samples tends to overestimate the species richness in the samples [RV11]. This is caused by several reasons, such as high sensitivity to sequencing errors, chimeric databases for taxonomy-dependent methods, and lab protocol-biases like primer biases and PCR biases [SID+15]. For these reasons shotgun sequencing is preferred over 16S rRNA sequencing if the goal is getting accurate estimation for the species richness and abundances at species-rank.

3.2 Taxonomy-dependent *versus* taxonomy-independent analysis

There are two main approaches for analysing the shotgun-metagenomics samples: *taxonomy-dependent* approach and *taxonomy-independent* approach [MMG12]. In taxonomy-dependent analysis, reads are mapped to a reference database of genomes, marker genes, proteins or some compositional models (e.g., HMMs models for the microbial genomes). Based on the mapping results, the richness and the abundances are estimated. This approach can be further split into three main methods: alignment-based, composition-based, and hybrid methods [MMG12]. We give a brief description of these methods and the most popular tools adopting them.

Alignment-based methods

In alignment-based methods, the reads are mapped to some reference database (e.g., NCBI bacterial database [TCF+14], Pfam protein database [FCE+16], or a curated database of phylogenetic marker genes), and the abundances are calculated based on the mapped reads. Different tools use different techniques for deciding the quality of the mapping, breaking the ties in mapping results, and finding the outliers. For example, MG-RAST server [MPD+08] uses BLAST to obtain an initial mapping, then it assigns the read to the best BLAST hit or to the lowest common ancestor of the

best BLAST hits. Other tools like MEGAN [HAQS07], SOrt-ITEMS [HGKM09], DiScRIBinATE [GHM10], MARTA [HBB10] and MetaPhyler [LGGP10] adopt a similar idea with different heuristic algorithms for deciding how to select the best hits and the taxonomic rank the reads should be assigned to. CARMA [KDG+08] uses BLASTx to map the reads to the Pfam protein sequences database, and then assigns the reads to a specific taxonomic rank based on the reconstruction of a phylogenetic tree of each matching Pfam family. MetaPhlAn [SWB+12], mOTU [SMZ+13], and GSMer [THZ14] map the reads to curated databases of marker regions (i.e., regions in the genomes that can be used to unambiguously identify different microbes at some taxonomy rank). MetaPhlAn uses BLAST or BowTie to map the reads to a curated database of clade-specific marker genes (a clade is a group of organisms that consists of a common ancestor and all its lineal descendants). It assigns the reads to a specific clade based on the mapping results, and estimates the relative abundance based on read counts and clades size. In mOTU, hidden Markov models (HMMs) are used for generating profiles for universal marker genes and for aligning the reads to those profiles, while in GSMer the reads are aligned to a database of genome-specific markers that are not restricted to coding regions.

Composition-based methods

Some genomic features can be used to distinguish between different organisms. For example, k-mer frequencies (the frequencies of all the possible subsequences of length k) and GC-content (the percentage of DNA nucleotides that are either G or C) differ among different organisms [Bir02]. Based on these features, compositional models can be built from a reference database, and the reads can be classified based on their similarity to these models. Different machine learning techniques are used to build these models and to classify the reads. For example, Phymm [BS09] creates interpolated Markov models (IMMs) for the bacterial species in NCBI RefSeq database, and maps the reads based on computed scores corresponding to the probability that they were generated by these IMMs. NBC [RRR11] builds k-mer frequency models from a reference database, and uses naive Bayes classifier to assign the reads to these models based on their log-likelihood scores. Phylopythia [MMT+07] uses support vector machine (SVM) to build SVM binary classifiers from the k-mer composition vectors of a reference database, then classifies the reads using these classifiers. TACOA [DKG+09] builds 4-mer frequency models, and uses a k-nearest neighbor algorithm to classify the sample reads.

Hybrid methods

In hybrid methods, a combination of alignment-based and composition-based methods is used to improve the quality and (or) the speed of reads classification. For example, PhymmBL [BS09] uses an equation which incorporates BLAST output and Phymm output to calculate the confidence of read alignments. SPHINX [MGSM11] utilizes a two-phases classification approach. In the first phase, it maps the reads to a 4-mer frequency model constructed from the reference database in order to reduce the search space. In the second phase, a SOrt-ITEMS-like algorithm is used to assign the reads to one of the candidate genomes chosen in phase one.

In taxonomy-independent analysis, there is no reference database, and usually an unsupervised machine learning algorithm is used for clustering the reads based on the similarity between them. For example, CompostBin [CYBE08] calculates the k-mer frequencies for various k, uses dimensionality reduction techniques to reduce the dimensionality of the compositional space, and then clusters the reduced vectors using clustering with normalized cut algorithm. MetaCluster [LYY+11] uses k-means algorithm for clustering the data, and provides a two-round binning algorithm, which separates the clustering of high-abundant species from the low-abundant ones. Other tools like TETRA [TWL+04], and AbundanceBin [WY11] use some other machine learning techniques for clustering the reads.

Both taxonomy-dependent and taxonomy-independent approaches have their own limitations. One major limitation in the taxonomy-dependent methods is the incompleteness of the reference databases. For example, only few thousands of bacterial species have been fully sequenced [LHJ+15] . This has two important impacts on the performance of the taxonomy-dependent tools. First, all tools fail to map a large number of the reads coming from the species that do not exist in the database (which we refer to as *unknown species*). Second a large number of reads coming from these unknown species can be falsely mapped to some species in the database due to their similarity with these species. On the other hand, most taxonomy-independent tools do not perform well on samples with low-abundance and very-low abundance species [WLYC12], and it is difficult to accurately decide the number of clusters (number of species in the sample) due to the high similarity between species in some genera. For the reasons above, getting accurate estimations for species richness and abundances at species-rank is difficult for both taxonomy-dependent and taxonomy-independent tools. As far as we know, there has been no performance comparisons between any taxonomy-dependent and taxonomy-independent tools,

hence we can not clearly say which approach might be better than the other.
Our tool can be classified as a taxonomy-dependent/alignment-based tool that works for shotgun sequencing. It provides accurate estimation for the richness and the abundances of the *known species* (species that exist in the reference database) at species-rank. In our work, we compare between our tool and some marker-based tools since they proved to achieve results better than the other tools.

3.3 Marker-based community profiling tools

Marker-based tools like MetaPhlAn and mOTU represent the current state-of-the-art for taxonomy-dependent metagenomics community profiling. They estimate the richness and the abundances at species-rank, they are very fast since they align the reads against very small databases, and they have lower false positive rates than other tools. However, the predicated abundances can be skewed, since they calculate the abundances based on the coverage inside small regions instead of the entire genome, and slight variations in the coverage in these regions can lead to skewed results. Due to certain biases in sequencing library preparations protocols and in sequencing machines, the coverage across the genome is not uniform [JHA+15, LKZR13]. For example, some regions might be highly covered, others might be low-covered, and some others might not be covered at all. Furthermore, the coverage results differ when we use different sequencing technologies [HDP+10].
The main idea behind our tool is to take into account the distribution of the reads across all genomes. By analyzing the coverage across entire genomes, we can filter out the outliers, and get a better estimation for the abundances, as shown in Figure 4.

4 MetaFlow: A taxonomy-dependent metagenomic community profiling tool

MetaFlow is a taxonomy-dependent metagenomic community profiling tool, which works for shotgun sequencing. It classifies the metagenomic reads at species taxonomic rank, and estimates the species richness and abundances in metagenomic samples.
The input to MetaFlow is a set of metagenomic reads, a database of reference genomes, and an initial mapping between the reads and the reference genomes (e.g.,

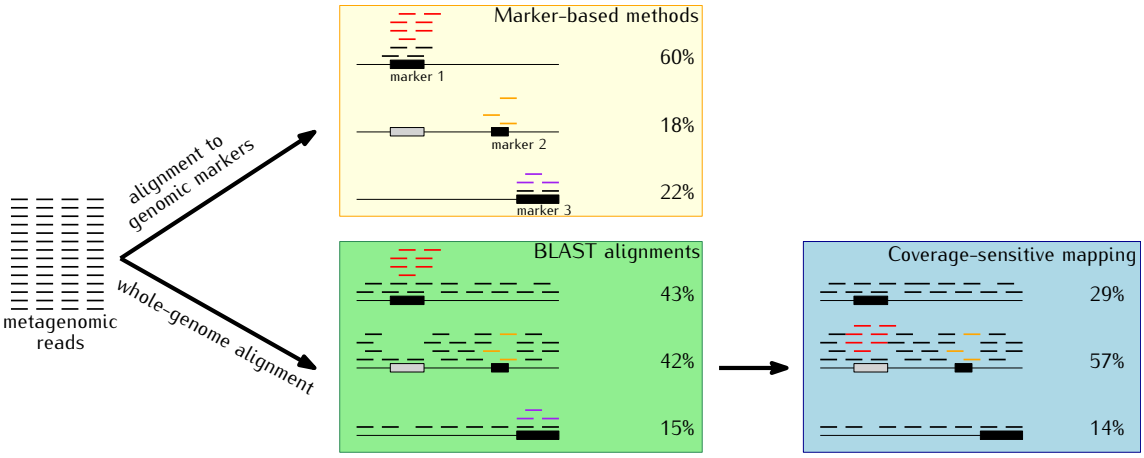


Figure 4: In the yellow box, reads are aligned only to genomic markers, and the relative abundances are highly skewed: marker 1 receives more false mappings (in red) because it is similar with a subsequence of the second genome (gray); marker 2 has a drop in coverage due to a sequencing bias, and it is covered only by three reads (orange); marker 3 is covered also by some reads sequenced from a species not in the reference database (violet). In the green box, reads have whole-genome BLAST alignments, but the relative abundances are still skewed: the tie in the alignment of the red reads (either to marker 1, or to the similar gray sequence in the second genome) is not resolved, and the violet reads from an unknown species aligning to the third genome are not removed. In the blue box, the coverage-sensitive mapping of the reads: the red reads are correctly aligned to the second genome (in the gray sequence) and the violet reads from an unknown species are discarded from the third genome. The abundances are relative to the known genomes [STM16].

a mapping obtained using BLAST). The output is an estimation of the richness and the relative abundances of the known species. This is obtained by assigning each read to exactly one reference genome, or marking it as originating from an unknown species (a species not present in the database). The optimal reads assignment is the one achieving the following three objectives: (I) for each reported genome, most regions are covered by some reads; (II) for each reported genome, the resulting read coverage across its regions is similar, up to some variation; (III) the total cost of assigning the reads to their genomes (we convert BLAST scores into costs) is minimized. Objective (I) detects outlier genomes (i.e., false positives). The idea is if a genome has only few regions covered, it is most likely to be an outlier, even if there is a large number of reads mapped to these regions. Such abnormal coverage does not comply with the shotgun sequencing assumption that most genomic

regions are sequenced if enough reads are sampled, and it happens most likely because genotypic similarity between different species [STM16]. Objective (II) is for breaking ties between read alignments with similar scores, and is also based on detecting abnormal read coverage patterns. For example, if most regions of a genome are covered, but some regions have abnormally high read coverage, then most likely also these coverage peaks are false alignments. This happens because these regions are similar to regions in the genomes from which the reads are truly sequenced, or because of sequencing biases [STM16]. Finally, objective (III) gives more confidence in selecting the correct read alignments.

4.1 Problem formulation and computational complexity

Our problem formulation is inspired by [LKZB13]. We formulate the problem of assigning the reads to their reference genomes as a minimum-cost flow problem on a bipartite graph, such that the reads form one part of the bipartition, the reference genomes form the other part, and BLAST mapping forms the arcs between them. The following detailed explanation of the problem formulation and the computational complexity are taken from our paper [STM16].

Assume that the reads have BLAST hits in the collection of reference genomes $\mathcal{G} = \{G^1, \ldots, G^m\}$. We partition every genome G^i into substrings of a fixed length L, which we call *chunks*. Denote by s_i the number of chunks that each genome G^i is partitioned into. We construct a bipartite graph $G = (A \cup B, E)$, such that the vertices of A correspond to reads, and the vertices of B correspond to the chunks of all genomes $G^1, \ldots, G^m$. Specifically, for every chunk j of genome G^i, we introduce a vertex y_j^i, and we add an edge between a read $x \in A$ and chunk $y_j^i \in B$ if there is a BLAST mapping of read x starting inside chunk j of genome G^i. This edge is assigned the cost of the mapping, which we denote here by $c(x, y_j^i)$. In order to model the fact that reads can originate from unknown species (whose genome is not present in the collection $\mathcal{G}$), we introduce an 'unknown' vertex z in B, with edges from every read $x \in A$ to z, and with a fixed cost $c(x, z) = \gamma$, where γ is appropriately initialized.

In the *coverage-sensitive metagenomic mapping* problem stated below, the tasks are: first, for each G^i, to find the optimal number of reads mapping to it, which we denote by r_i (which is 0 if G^i is an outlier); second, to select an optimal subset $M \subseteq E$ such that for every read $x \in A$ there is exactly one edge in M covering it (a mapping of x). Objectives (I)-(III) guide these two tasks, as follows. Objective (I)

will be achieved by filtering out outlier genomes from the graph at each iteration. Objectives (II) and (III) are achieved by minimizing the sum of the following two costs:

(A) the sum, over all chunks of every genome G^i, of the absolute difference between r_i/s_i and the number of read mappings it receives from M (Objective (II));

(B) the sum of all edge costs in M (Objective (III)).

Our formal problem definition is given below. We use the following notation: n is the number of reads, m is the number of different genomes where the reads have BLAST hits, s_i is the number of chunks of each genome G^i, $i \in \{1, \ldots, m\}$, and in a graph $G = (V, E)$, $d_M(v)$ denotes the number of edges of a set $M \subseteq E$ incident to a vertex v.

Coverage sensitive metagenomic mapping:
Input:

- a bipartite graph $G = (A \cup B, E)$, where A is the set of n reads, $B = \{y_1^1, \ldots, y_{s_1}^1, \ldots, y_1^m, \ldots, y_{s_m}^m\} \cup \{z\}$ is the set of all genome chunks plus z, the 'dummy' node,
- a cost function $c : E \to \mathbb{Q}$,
- constants $\alpha \in (0, 1)$, $\beta, \gamma \in \mathbb{Q}_+$.

Tasks:

- find a vector $R = (r_1, \ldots, r_m)$ containing the number of reads mapping to each genome G^i, $i \in \{1, \ldots, m\}$,
- find a subset $M \subseteq E$ such that $d_M(x) = 1$ holds for every $x \in A$ (i.e., each read is covered by exactly one edge of M)

which together minimize:

$$(1-\alpha)\sum_{\{x,y\}\in M} c(x,y) + \\ + \alpha \cdot \beta \cdot \sum_{i=1}^{m}\sum_{j=1}^{s_i}\left|\frac{r_i}{s_i} - d_M(y_j^i)\right| + \\ + \gamma d_M(z).$$

Theorem 4.1.1. [STM16] *The coverage-sensitive metagenomic mapping problem is NP-hard for all* $\alpha \in (0, 1)$.

Proof. We reduce from the exact cover with 3-sets (X3C) problem. In this problem, we are given a collection $\mathcal{S}$ of 3-element subsets $S_1, \ldots, S_m$ of a set $U = \{1, \ldots, n\}$, and we are required to decide if there is a subset $\mathcal{C} \subseteq \mathcal{S}$, such that every element of U belongs to exactly one $S_i \in \mathcal{C}$.

Given an instance of Problem X3C, we construct the bipartite graph $G = (A \cup B, E)$, where $A = U = \{1, \ldots, n\}$ and B corresponds to $\mathcal{S}$, in the following sense:

- $s_i = 3$, for every $i \in \{1, \ldots, m\}$
- for every $S_j = \{i_1 < i_2 < i_3\}$, we add to B the three vertices $y_{j,1}, y_{j,2}, y_{j,3}$ and the edges $\{i_1, y_{j,1}\}$, $\{i_2, y_{j,2}\}$, $\{i_3, y_{j,3}\}$, each with cost 0.

For completeness, we also add vertex z to B, and edges of some positive cost between it and every vertex of A.

We now show that for any $\alpha \in (0, 1)$, an instance for Problem X3C is a 'yes' instance if and only if the coverage-sensitive metagenomic mapping problem admits on this input a solution set $M \subseteq E$ of cost 0. Observe that in any solution M of cost 0, the genome abundances are either 0 or 3, and that vertex z has no incident edges in M.

For the forward implication, let $\mathcal{C}$ be an exact cover with 3-sets. We assign the abundances $c_1, \ldots, c_m$ as follows:

$$c_j = \begin{cases} 3, & \text{if } S_j \in \mathcal{C} \\ 0, & \text{if } S_j \notin \mathcal{C} \end{cases}$$

and we construct M as containing, for every $S_j = \{i_1 < i_2 < i_3\} \in \mathcal{C}$, the three edges $\{i_1, y_{j,1}\}$, $\{i_2, y_{j,2}\}$, $\{i_3, y_{j,3}\}$. Clearly, this M gives a solution of cost 0 to the coverage-sensitive mapping and abundance estimation problem.

Vice versa, if the coverage-sensitive mapping and abundance estimation problem admits a solution M of cost 0, in which the genome coverage are either 0 or 3 and z has no incident edges, then we can construct an exact cover with 3-sets $\mathcal{C}$ by taking:

$$\mathcal{C} = \{S_i \mid d_M(y_{i,1}) = d_M(y_{i,2}) = d_M(y_{i,3}) = 1\}.$$

□

Due to the hardness result from Theorem 4.1.1, we opt for the common *iterative refinement strategy*, akin to the strategy behind k-means clustering [Ste56], or Viterbi training strategies with Hidden Markov Models [DEKM98]. In Section 4.3 we present the details of this approach, but the main ideas are:

1. If the unknown vector $R = (r_1, \ldots, r_m)$ is fixed to some value $R = (a_1, \ldots, a_m)$, then finding only the optimal mapping M can be solved in polynomial-time with minimum-cost flows. We show this in Section 4.2.

2. For finding the optimal R, we start with a vector $R^0 = (a_1, \ldots, a_m)$, where a_i equals the number of reads with BLAST hits to G^i. We repeat the following process, until the vector R converges to a stable value. For each iteration j:

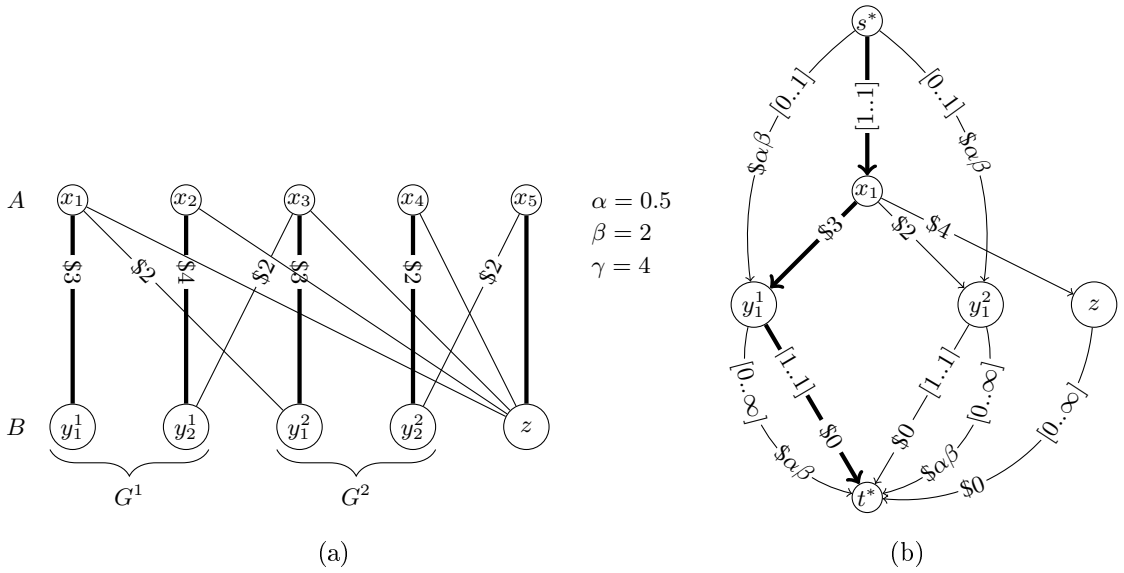
   (a) compute the optimal mapping M^j with minimum-cost flows, using R^j as input;

   (b) update R^j to R^{j+1}, a vector whose i-th component equals $mean_i \cdot s_i$; here $mean_i$ is the 20%-trimmed mean read coverage of the chunks of genome G^i, obtained from M^j.

4.2 The reduction to a minimum-cost flow problem

The reduction to minimum-cost flow is taken from our paper [STM16]. Given an input graph $G = (A \cup B, E)$ for the coverage-sensitive metagenomic mapping problem, with a fixed vector $R^j = (a_1, \ldots, a_m)$ of read mappings to the m genomes (at an iteration j), we construct a flow network $N = (G^*, \ell, u, c, q)$ (see Figure 5(b)). Here, $\ell, u : E(G^*) \to \mathbb{N}$ are the *demand* and *capacity* of every arc, respectively; $c : E(G^*) \to \mathbb{Q}$ is the function giving the cost per unit of flow for every arc of G^*; q is the required value of the flow, that is, the value of the flow that must exit the unique source of G^*, which equals the value of the flow that must enter the unique sink of G^*.

The construction of G^* starts with G, and orients all its edges from A toward B, and sets their demand to 0, their capacity to 1, and their cost as in the input metagenomic mapping instance, multiplied by $(1 - \alpha)$. We add a global source s^* with arcs toward all vertices in A, with demand and capacity 1, and 0 cost. Since we pay the fixed penalty $\alpha \cdot \beta$ for every read below or above the given coverage of each genome chunk y^i_j, we add the following arcs:



(a) (b)

Figure 5: Figure 5(a): A bipartite graph $G = (A \cup B, E)$, whose edges are labeled with costs. The costs of the edges incident to z are not drawn, and they all equal $\gamma = 4$. An optimal solution for the coverage-sensitive metagenomic mapping is shown (highlighted edges). The optimal read mappings to G^1 and G^2 are $r_1 = r_2 = 2$, and the optimal value of the objective function is $(1-\alpha)12+\alpha\beta(1-1+1-1+1-1+1-1)+\gamma = 10$. Figure 5(b): A subpart of the flow network N constructed from G, corresponding to the vertices x_1, y_1^1, y_1^2. The arcs with flow value 1 are highlighted. The demands and capacities of an arc (u, v) are indicated as $[\ell(u, v)..u(u, v)]$ and its cost as $\$c(u, v)$.

- an arc from y_j^i to a global sink t^* with $\ell(y_j^i, t^*) = u(y_j^i, t^*) = a_i/s_i$, and $c(y_j^i, t^*) = 0$;
- an arc from y_j^i to t^* with $\ell(y_j^i, t^*) = 0$, $u(y_j^i, t^*) = \infty$, and $c(y_j^i, t^*) = \alpha \cdot \beta$;
- an arc from s^* to y_j^i with $\ell(s^*, y_j^i) = 0$, $u(s^*, y_j^i) = a_i/s_i$ and $c(s^*, y_j^i) = \alpha \cdot \beta$.

We also add arcs from every vertex $x \in A$ to the unknown vertex $z \in B$, with demand $\ell(x, z) = 0$, $u(x, z) = 1$, and cost $c(x, z) = \gamma$. From z we add an arc to t^* with $\ell(z, t^*) = 0$, $u(z, t^*) = \infty$, $c(z, t^*) = 0$.

Finally, we set $q = n+\sum_{i=1}^{m} a_i$, and also add the arc (s^*, t^*) with 0 demand and cost, and infinite capacity. The optimal matching for the coverage-sensitive metagenomic mapping problem consists of the edges of G whose corresponding arcs in G^* have non-zero flow value in the integer-valued minimum-cost flow over N. See Figure 5 for an example.

The correctness of this reduction can be seen as follows. As long as a genome chunk y_j^i has exactly a_i/s_i reads mapped to it, then no cost is incurred; this is represented by the arc (y_j^i, t^*) with demand and capacity a_i/s_i, and 0 cost. If y_j^i receives more reads than a_i/s_i, then these additional reads will flow along the parallel arc (y_j^i, t^*) of cost $\alpha \cdot \beta$. If y_j^i receives less reads than a_i/s_i, then some compensatory flow comes from s^* through the arc (s^*, y_j^i), where these fictitious reads again incur cost $\alpha \cdot \beta$. We set the capacity of (s^*, y_j^i) to a_i/s_i since at most a_i/s_i fictitious reads are required to account for the lack of reads for genome chunk y_j^i.

Requiring flow value $q = n + \sum_{i=1}^{m} a_i$ ensures that there is enough compensatory flow to satisfy the demands of the arcs of type (y_j^i, t^*). Having added the arc (s^*, t^*) with 0 cost ensures that there is a way for the exogenous flow not needed to satisfy demand constraints to go to t^*, without incurring an additional cost.

4.3 Implementation details

Our practical implementation is divided into five stages. They depend on some parameters explained in Appendix 1, where also the input and output format, and other features of the tool are explained. The five stages are explained below, and are shown in Figure 6.

Stage 1: Removing outlier species
In this stage, we remove each outlier genome and the reads that have BLAST hits only to it. A genome $G^i \in \mathcal{G}$ is considered an outlier if at least one of the following conditions holds.

- The average read coverage of G^i (i.e., the number of reads with BLAST hits to G^i multiplied by the average read length and divided by the length of G^i) is lower than a given parameter.
- The average read mapping per chunk (i.e., the number of reads with BLAST hits to G^i divided by s_i) is lower than a given parameter.
- The percentage of chunks without any BLAST hit is more than a given parameter.

Stage 2: Breaking ties inside each genome
A read can have BLAST hits to different chunks of the same genome. In this stage, for each read remaining after Stage 1, we select only one BLAST hit in each genome,

as follows. For each remaining genome $G^i \in \mathcal{G}$, we create a sub-problem instance $G = (A \cup B, E)$ where A consists only of the reads that have BLAST hits to G^i, and B consists only of the chunks of G^i (excluding the unknown vertex z, which will be dealt with in Stage 4). We fix the one-element vector R of read mappings as $|A|$, and solve this sub-problem using the minimum-cost flow reduction described in Section 4.2. After this stage, every read has at most one hit to each genome, but it can still have hits to multiple genomes.

Stage 3: Breaking ties across all genomes
A read can be mapped to different species, due to the similarity between their genomes. In order to select only one read mapping across all genomes, we solve the coverage-sensitive metagenomic mapping problem on a graph $G = (A \cup B, E)$, as follows. The set A consists of all remaining reads, and the set B of all chunks of the remaining genomes. The set of edges E is the one obtained by the filtration done in Stage 2. Since this problem is NP-hard, we employ the iterative refinement strategy, coupled with minimum-cost flows, mentioned at the end of the previous section. After each iteration j, we use the resulting mapping M^j to remove outlier genomes, as in Stage 1. After this stage, each read is mapped to exactly one genome, and to only one of its chunks.

Stage 4: Identifying reads from unknown genomes
In this stage we identify reads originating from species whose reference genomes are not present in the reference database. We run the same minimum-cost flow reduction as in Stage 2, to which we add the unknown vertex z. If a read is mapped to z, then it will be marked as coming from an unknown genome and removed from the graph. Finally, we again remove outlier genomes, as in Stage 1.

Stage 5: Estimating richness and relative abundances
For every genome G^i, we compute its average read coverage $read\_coverage(i)$, and its relative abundance $rel\_abundance(i)$ as: $read\_coverage(i) = r_i \cdot R/length(i)$, $rel\_abundance(i) = read\_coverage(i)/\sum_{j=1}^{m} read\_coverage(j)$, where r_i is the number of reads mapping to G^i after Stages 1-4, R is the average read length, and $length(i)$ is the length of G^i.

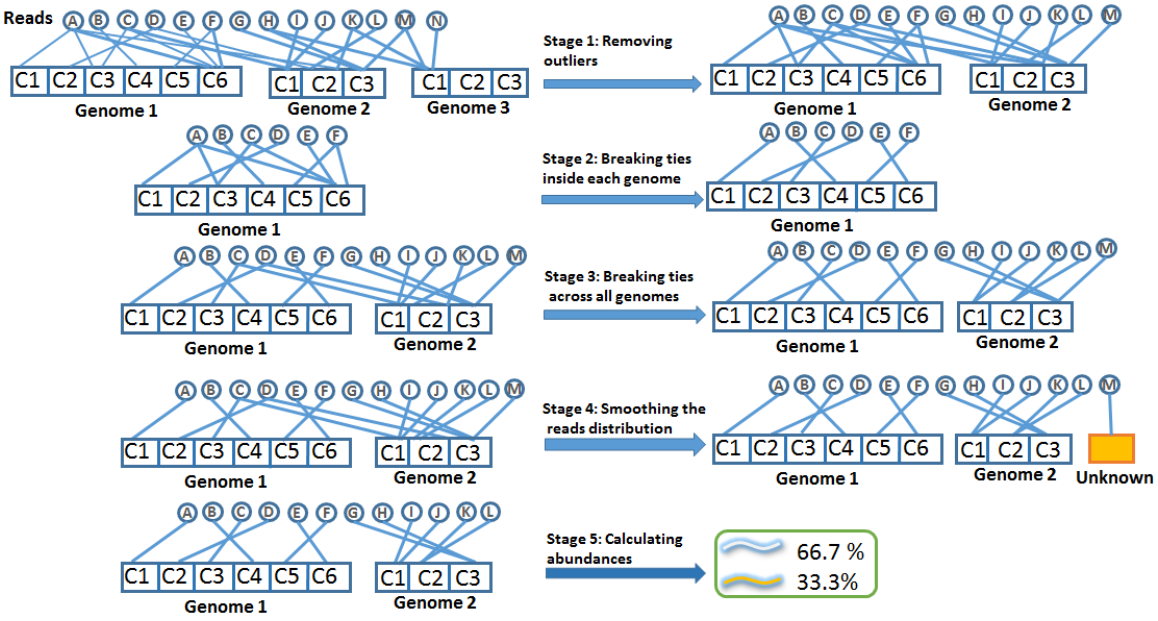


Figure 6: MetaFlow stages

5 Experiments and results

In order to evaluate the performance of MetaFlow, we ran a set of extensive experiments on simulated metagenomics data, in addition to one real metagenomics dataset, and we compared our results against BLAST, MetaPhlAn, mOTU and GSMer.

5.1 Experimental setup

Synthetic datasets

The simulated data were created using Metasim [ROA+08]. We used 817 bacterial genomes extracted from the NCBI microbial genome database [PBH+14], from which we created 48 different datasets using the default 454-pyrosequencing error model (with mean=250 and standard deviation=1). The species abundances inside each simulated dataset followed a log-normal distribution with mean=1, and standard deviation=1. The 48 datasets were divided into two groups: a) Low-complexity datasets (LC), which had 30 datasets, each one of them contained 4 million reads sampled from 15 different species, and b) High-complexity datasets (HC), which had 18 datasets, each one of them contained 40 million reads sampled from 100 species. The complexity here refers to the number of species in the sample. Our experimental

setup was in line with the previous study [SWB+12]. We used the 48 datasets to run 4 different experiments, which we called LC-Known, HC-Known, HC-Known and HC-Unknown. These experiments were designed to separately measure the effect of the following on performance of the competing tools: 1) the degree of genotypic similarity between microbial species, 2) incomplete databases, and 3) the complexity of the microbial samples.

LC-Known: The goal of the LC-Known experiment was to evaluate the performance of MetaFlow in situations where all sample's species exist in the reference database. We call these species *known species*. This perfect information scenario could be an unrealistic situation in metagenomics analysis, nonetheless, we used it as a lower bar for the comparison between all tools. In addition, we wanted to measure the effect of genotypic similarity between microbial species on the results. Similarity between bacterial species is one of the reasons that metagenomics profiling is a challenging task. The higher the degree of similarity between the species, the more difficult it is to find the correct mapping, and to filter the outliers out. In this experiment, we used 15 LC datasets. These datasets were constructed by varying the degree of genotypic similarity between the species selected in each dataset, which we measured using the genomic-distance index given by MUMmer [DPCS02]. Table 1 covers the LC datasets in more details.

LC-Unknown: In this experiment we evaluated the performance of MetaFlow in situations where some sample's species do not exist in the reference database, which is closer to a real-life scenario. At the moment, the microbial databases are far from being complete, and we are capturing only a very tiny piece of the microbial species diversity. The 15 LC datasets used before in LC-Known were also used in this experiment with one modification. For each dataset, we randomly chose 3 species, and replaced them with another 3 species that did not exist in our reference database. These species were selected from the NCBI microbial genome database published after 2014, and we called them *unknown species*.

HC-Known: This also was a perfect information scenario. Our objective here was to evaluate the robustness of the tool when the complexity increases. For that, we used 9 HC datasets. The species in first eight datasets HC1:HC8 were randomly chosen, while the species in the last dataset HC9 were selected from 15 genera with different degree of similarity between their species.

HC-Unknown: This is supposed to be the closest to a realistic scenario in metagenomics analysis (high complexity, large number of reads, and the sample contains unknown species). To construct the HC-Unknown datasets, we took the HC-Known

datasets, randomly chose 15 species from each dataset, and replaced them with another 15 from the unknown species.

Real metagenomics data

We also tested MetaFlow using real metagenomics data. For that, we merged 6 G_DNA_Stool samples of a female from the Human Microbiome Project (accession numbers SRX877403, SRX877395, SRX877412, SRX024197, SRX025210 and SRX877173). Five samples were generated using Illumina, and one sample using LS454. The read length was normalized to 100bps for all reads. The total number of reads from all samples was 287,565,377, out of which 98,223,162 BLAST mapped to one or more species. Only alignments with identity $\geq 97\%$ were selected as an input for MetaFlow. In addition to the full reference genomes in NCBI's bacterial database, we added two other reference genomes: a supercontig of *B.uniformis* (accession number NZ_JH724260.1), because *B.uniformis* was previously reported as most abundant in fecal samples [QLR$^+$nt]; and the longest scaffold of *B.plebeius* (accession number NZ_DS990130.1) because MetaPhlAn and mOTU reported *B.plebeius* as most abundant in this sample.

Evaluation method

We used the following measures in order to evaluate the goodness of the richness estimations: *sensitivity* (true positive rate), *precision*, and the *harmonic mean* of sensitivity and precision. The harmonic mean is also called *F-score*. To evaluate the goodness of the relative abundances prediction, we measured the ℓ_1-*norm* of between the true and predicted abundances. Table 2 shows the definition of these measures.

We compared the performance of our tool against GSMer [THZ14], mOTU [SMZ$^+$13], MetaPhlAn [SWB$^+$12] and BLAST [AGM$^+$90]. In BLAST analysis, we always selected the best alignment. In case of multiple equally good alignments, we randomly selected one of them; species with coverage ≤ 0.3 were considered outliers and removed from the results. GSMer does not provide relative abundances, so we compared only the accuracy of the richness estimations. For mOTU, some known species did not exist in its database, so in our evaluation it received full marks on these species. On the other hand, some of the unknown species (in LC-Unknown and HC-Unknown experiments) already existed in mOTU's database. For these species, we removed mOTU's correct prediction.

Dataset	Similarity	Description
LC1, LC2, LC3, LC4	Very low-similarity	Each species was selected from a different genus.
LC5, LC6, LC7, LC8	Very low-similarity	Randomly selecting all species, two species might come from same genus.
LC9	Low-similarity	1 species from *Lactobacillus* genus, and the rest randomly selected.
LC10	Low-similarity	3 species from *Lactobacillus* genus, and the rest randomly selected.
LC11	Medium-similarity	1 species from *Shigella* genus, and the rest randomly selected.
LC12	Medium-similarity	3 species from *Shigella* genus, and the rest randomly selected.
LC13	High-similarity	1 species from *Brucella* genus, and the rest randomly selected.
LC14	High-similarity	3 species from *Brucella* genus, and the rest randomly selected.
LC15	High-similarity	Randomly selecting 15 species from only 3 genera, *Streptococcus*, *Brucella* and *Bacteroides*.

Table 1: LC-known datasets

5.2 Results

Synthetic datasets

Figure 7 shows a comparison between the sensitivity and accuracy results for the simulated data, and Table 3 shows the average F-score and ℓ_1-norm results for the same data. As shown in Figure 7, BLAST achieves the best score for the sensitivity measure. The result is expected since BLAST reports all species with good alignments. However, this comes with a big decrease in precision, because BLAST does not have a strategy for removing outliers or breaking the ties between alignments with similar scores. Both MetaPhlAn and mOTU have better precision and F-score than BLAST, which confirms that using marker regions is a good strategy for distinguishing between similar species and removing outliers. MetaFlow achieves the best precision and F-score which confirms our hypothesis that considering the coverage across the entire genome gives more accurate estimation than

Evaluation Measures
$sensitivity = \frac{number\ of\ correctly\ identified\ species}{actual\ number\ of\ species\ in\ the\ sample}$
$precision = \frac{number\ of\ correctly\ identified\ species}{number\ of\ predicted\ species}$
$F\text{-}score = 2 \cdot \frac{sensitivity \cdot precision}{sensitivity + precision}$
$\ell_1\text{-}norm = \sum_{i=1}^{k} \lvert True\ abundance_i - Predicted\ abundance_i \rvert$, $for\ i \in \{true\ species\ in\ the\ sample \cup predicted\ species\}$

Table 2: Evaluation measures for the results

marker-based methods. This is explained by the fact that marker regions are much shorter compared to the genome length, and any slight variations in coverage in these regions can easily skew the abundance estimation. On the simulated data MetaFlow gives a much better abundance estimation than marker-based methods, with an improvement of 2-4× in average ℓ_1-norm. Using our strategy for filtering out outlier species and false alignments, MetaFlow also achieves better abundance estimation than BLAST. The variance in sensitivity of the marker-based methods in our experiments suggests that such an approach is not always accurate in identifying all species present in the sample. For example, MetaPhlAn's sensitivity is not maximum in the LC12,LC13,LC14-Known datasets, even though these are perfect-information scenarios.

The results also show that MetaFlow is robust with the increase in sample size. The same good results are also obtained in the HC scenario. Finally, the datasets with high genotypic homogeneity (LC13-LC15) show that this scenario remains a difficult one: even though MetaFlow improves both the richness and abundance estimation over the competing methods, its precision drops to an average of 0.85 and its ℓ_1-norm increases to an average of 40. We give the complete results for the LC-Known and LC-Unknown experiments in Appendix 2.

Finally, the results are better on the HC datasets than on the LC datasets for all tools. The reason is that LC datasets were constructed by including different level of similarity between their species. On the other hand HC datasets were all selected randomly except HC9 dataset. Also LC-Unknown datasets had 20% unknown species, while HC-Unknown had only 15%. In addition, since the number of species in LC datasets in much smaller than in HC dataset, errors in predictions

are more likely to be more severe in LC datasets than HC datasets.

Real dataset

Figure 8 shows the phylogenetic tree for the species found by MetaFlow in the real sample. The most abundant species reported by MetaFlow is *B.uniformis* (23.6% relative abundance), which was also reported as the most abundant species in human gut [QLR$^+$nt]. This is also supported by the fact that 15,418,699 reads are mapped by BLAST only to *B.uniformis*. Out of these, 10,721,492 are finally assigned by MetaFlow to *B.uniformis*, because of the uneven read coverage. This corresponds to an average read coverage of 220. To put this number into perspective, the 10th most abundant species according to MetaFlow, *A.shahii*, has relative abundance 2.3% and average read coverage 21. MetaPhlAn and mOTU assign to *B.uniformis* abundances 1.7% and 6.4%, respectively. The second most abundant species reported by MetaFlow is *B.vulgatus*, another common species in human gut [QLR$^+$nt]. MetaFlow's predicted abundance is 22.3% (average read coverage 208), which is in line with MetaPhlAn's prediction of 17.7% and, to an extent, mOTU's prediction of 11.9%. In Table 5 we give the list of the top 10 prediction of MetaFlow, and their abundances reported by MetaPhlAn and mOTU. Four species out of the top six species have also been reported by [QLR$^+$nt] as predominant in human gut, and they constitute 59% of the sample according to MetaFlow (relative to the species known to MetaFlow in this experiment). It is important to note that our abundance calculation is relative to the known genomes only, and 62 of the species reported by MetaPhlAn are not available in our BLAST database. As a consequence, the actual relative abundance for *B.uniformis* might be lower than 23.6%, however, it cannot be significantly lower than *B.vulgatus*, as MetaPhlAn and mOTU predicted.

Running time

As shown in Table 4, the running time for MetaPhlAn, mOTU and GSMer is better than MetaFlow because they use a small curated databases of markers, while MetaFlow uses the complete genomes. One way to improve the running time of MetaFlow is to randomly select small regions of the microbial genomes instead of using the complete genomes, however we have not tested this method.

6 Conclusion

Microbes play an indispensable role in our life, and it is essential for us to understand their world. Metagenomics proved to be a powerful tool for unveiling the secrets

	LC-Known		LC-Unknown	
	F-score	ℓ_1-norm	F-score	ℓ_1-norm
MetaFlow	**0.971**	**10.41**	**0.825**	**17.87**
MetaPhlAn	0.946	26.42	0.770	31.48
BLAST	0.909	12.46	0.745	21.47
GSMer	0.218	N/A	0.163	N/A
mOTU	0.924	36.31	0.780	43.74
	HC-Known		HC-Unknown	
	F-score	ℓ_1-norm	F-score	ℓ_1-norm
MetaFlow	**0.976**	**4.86**	**0.883**	**8.01**
MetaPhlAn	0.958	18.61	0.844	19.13
BLAST	0.920	5.94	0.809	11.25
GSMer	0.327	N/A	0.259	N/A
mOTU	0.949	10.55	0.847	18.73

Table 3: Average over the F-score and ℓ_1-norm in each experimental scenario. The 15 LC datasets contain 4M reads from 15 species, and the 9 HC datasets contain 40M reads from 100 species.

of the microbial world. Metagenomics community profiling, which is estimating the richness and the abundances of the species in metagenomics samples, is a crucial step in metagenomics analysis. Due to some limitations in the current sequencing technology, high similarity between microbial species and the incompleteness of the microbial databases, metagenomics community profiling remains a challenging problem. At the time being, it is hard to accurately estimate the richness and the abundances of all the species inside metagenomics samples, and what we can opt for is to reveal part of the complexity inside the samples. There are different tools for solving the metagenomics community profiling problem and they have their own advantages and drawbacks. The current state-of-the-art for taxonomy-dependent community profiling is to use a curated database of markers which can uniquely identify microbes at species taxonomic rank. Marker-based methods calculate the abundances based on small regions instead of the entire genome, and slight variations in the coverage in these regions can lead to skewed results. We showed that taking into account the distribution of reads across the whole genome landscape provides more information which can improve the results over the marker-based methods. We introduced MetaFlow, a new tool for solving the metagenomics community profiling problem using high-throughput sequencing data. We also proved that coverage-

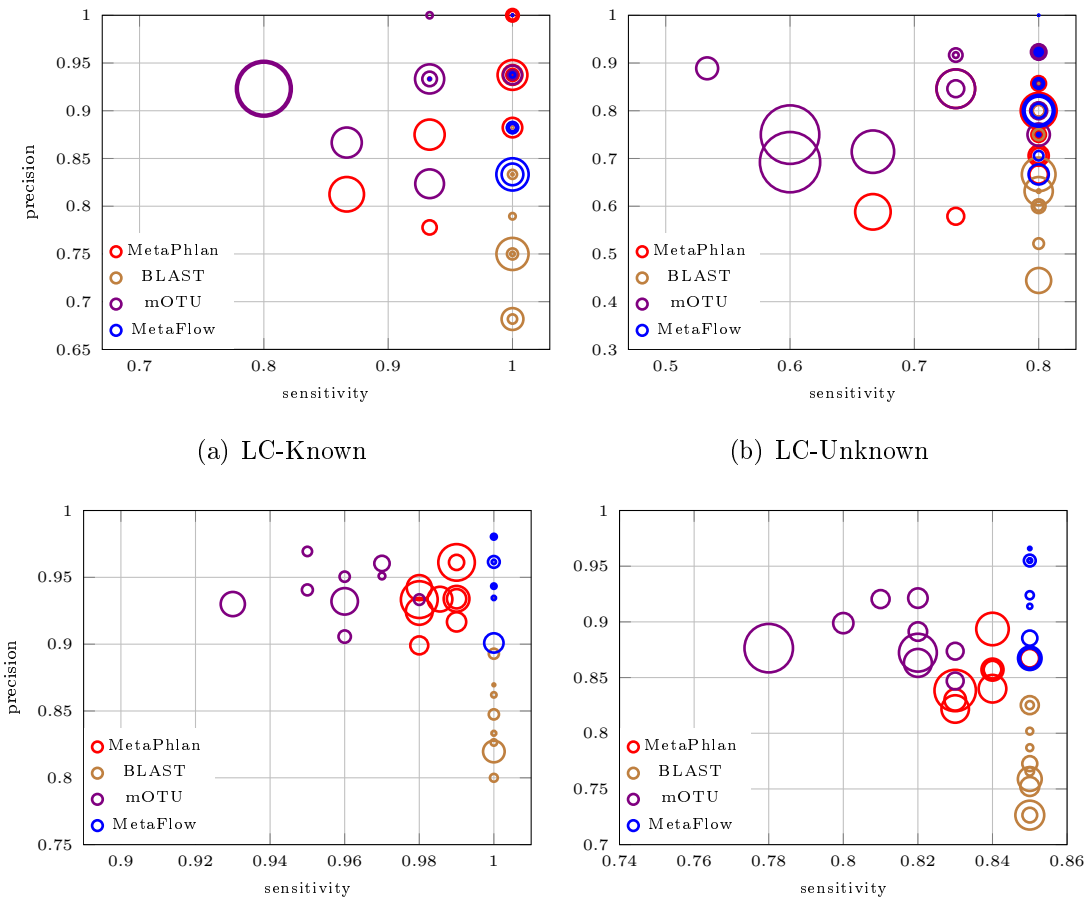
	4M reads	40M reads	280M reads
MetaFlow	28	459	2025
MetaPhlAn	14	132	387
BLAST	243	1572	3696
GSMer	42	364	N/A
mOTU	**9**	**84**	**380**

Table 4: Average running time in minutes in each experimental scenario.

	MCF	**MetaPhlAn**	**mOTU**
Bacteroides uniformis	0.269	0.026	0.140
Bacteroides vulgatus	0.255	0.260	0.261
Eubacterium rectale	0.118	0.136	0.082
Bacteroides xylanisolvens	0.105	0.061	0.066
Bacteroides plebeius	0.059	0.365	0.344
Bacteroides thetaiotaomicron	0.057	0.002	0.011
Faecalibacterium prausnitzii	0.042	0.056	0.018
Parabacteroides distasonis	0.039	0.001	0.027
Akkermansia muciniphila	0.032	0.061	0.032
Alistipes shahii	0.026	0.030	0.020

Table 5: A comparison between the relative abundance of the top 10 species reported by MetaFlow and their relative abundance as reported by MetaPhlAn and mOTU (normalized by the sum of relative abudances of these 10 species). *B.uniformis* is reported as the most abundant species by MetaFlow, which is aligned with [QLR$^+$nt], while it was reported as low abundant by MetaPhlAn and mOTU.

sensitive metagenomic mapping problem is NP-hard, and showed how network flows can be used for solving the problem. Finally, for metagenomics community profiling based on reference databases, perhaps a mixture between coverage-sensitive metagenomic mapping and marker-based methods can lead to even better results.



(a) LC-Known

(b) LC-Unknown

(c) HC-Known

(d) HC-Unknown

Figure 7: Results of the tools on all simulated datasets. The x-axis is the sensitivity and the y-axis is the precision; each circle is one experiment; inside each plot, the size of the circles is proportional with the ℓ_1-norm (smaller is better). In LC-Known dataset: 15 datasets, each with 4M reads from 15 species, all known. In LC-Unknown: 15 datasets, each with 4M reads from 15 species, out of which 3 are unknown. In HC-Known: 9 datasets, each with 40M reads from 100 species, all known. In HC-Unknown: 9 datasets, each with 40M reads from 100 species, out of which 15 are unknown. The unknown species are among 31 bacterial species from the NCBI microbial genome database, published after 2014. GSMer's results are not included in the figures, since its precision and sensitivity were always much lower than the other methods (see Table 3).

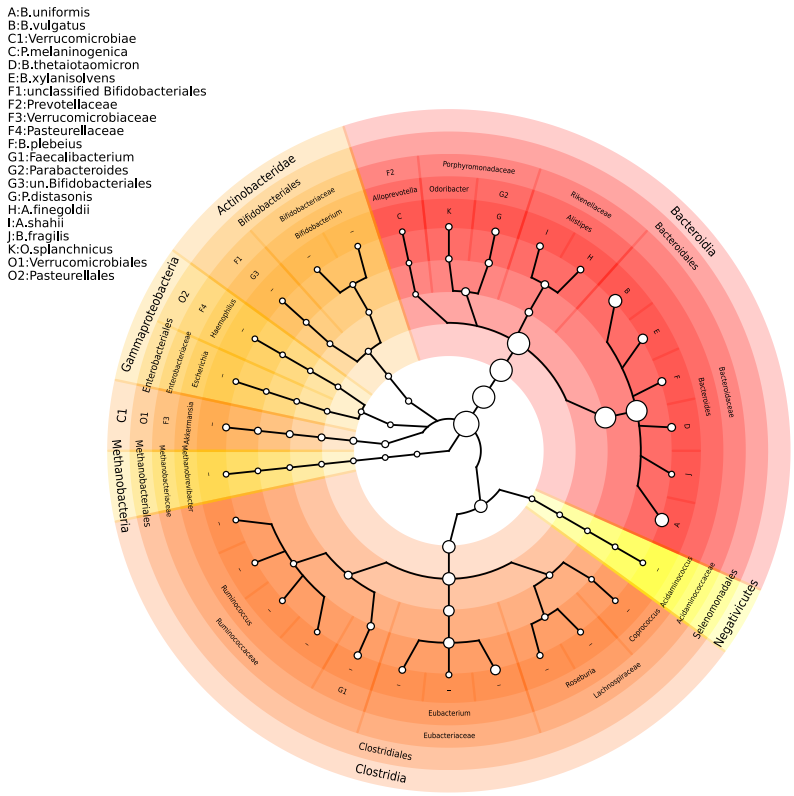


Figure 8: The species found by MetaFlow in the real sample obtained by merging 6 G_DNA_Stool samples from the Human Microbiome Project (287M reads out of which 57M BLAST mapped to some species). The size of the circles is proportional to the relative abundance at that taxonomic level. NCBI's bacterial database of reference bacterial genomes was used by MetaFlow.

References

AGM+90 Altschul, S. F., Gish, W., Miller, W., Myers, E. W. and Lipman, D. J., Basic local alignment search tool. *Journal of Molecular Biology*, 215,3(1990), pages 403–410. URL `http://dx.doi.org/10.1006/jmbi.1990.9999`.

ALS95 Amann, R. I., Ludwig, W. and Schleifer, K.-H., Phylogenetic identification and in situ detection of individual microbial cells without cultivation. *Microbiological Reviews*, 59,1(1995), pages 143–169.

BDW+04 Bäckhed, F., Ding, H., Wang, T., Hooper, L. V., Koh, G. Y., Nagy, A., Semenkovich, C. F. and Gordon, J. I., The gut microbiota as an environmental factor that regulates fat storage. *Proceedings of the National Academy of Sciences of the United States of America*, 101,44(2004), pages 15718–15723. URL `http://www.pnas.org/content/101/44/15718.abstract`.

Bir02 Birdsell, J. A., Integrating genomics, bioinformatics, and classical genetics to study the effects of recombination on genome evolution. *Molecular Biology and Evolution*, 19,7(2002), pages 1181–1197.

BS09 Brady Arthur and Salzberg Steven L, Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models. *Nature Methods*, 6,9(2009), pages 673–676. URL `http://www.nature.com/nmeth/journal/v6/n9/suppinfo/nmeth.1358\_S1.html`. 10.1038/nmeth.1358.

CYBE08 Chatterji, S., Yamazaki, I., Bai, Z. and Eisen, J. A., CompostBin: A DNA composition-based algorithm for binning environmental shotgun reads. *Research in Computational Molecular Biology*. Springer, 2008, pages 17–28.

DEKM98 Durbin, R., Eddy, S. R., Krogh, A. and Mitchison, G., *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

DJK11 Dezs, B., Jüttner, A. and Kovács, P., Lemon - an open source C++ Graph Template Library. *Electronic Notes in Theoretical Computer Sci-*

ence, 264,5(2011), pages 23–45. URL `http://dx.doi.org/10.1016/j.entcs.2011.06.003`.

DKG+09 Diaz, N. N., Krause, L., Goesmann, A., Niehaus, K. and Nattkemper, T. W., TACOA–Taxonomic classification of environmental genomic fragments using a kernelized nearest neighbor approach. *BMC Bioinformatics*, 10,1(2009), page 56.

DPCS02 Delcher, A. L., Phillippy, A., Carlton, J. and Salzberg, S. L., Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Research*, 30,11(2002), pages 2478–2483.

FCD+76 Fiers, W., Contreras, R., Duerinck, F., Haegeman, G., Iserentant, D., Merregaert, J., Min Jou, W., Molemans, F., Raeymaekers, A., Van den Berghe, A. et al., Complete nucleotide sequence of bacteriophage MS2 RNA: primary and secondary structure of the replicase gene. *Nature*, 260,5551(1976), pages 500–507.

FCE+16 Finn, R. D., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Mistry, J., Mitchell, A. L., Potter, S. C., Punta, M., Qureshi, M., Sangrador-Vegas, A. et al., The Pfam protein families database: towards a more sustainable future. *Nucleic Acids Research*, 44,D1(2016), pages D279–D285.

GHM10 Ghosh, T. S., Haque, M. and Mande, S. S., DiScRIBinATE: a rapid method for accurate taxonomic classification of metagenomic sequences. *BMC Bioinformatics*, 11,Suppl 7(2010), page S14.

GPD+06 Gill, S. R., Pop, M., DeBoy, R. T., Eckburg, P. B., Turnbaugh, P. J., Samuel, B. S., Gordon, J. I., Relman, D. A., Fraser-Liggett, C. M. and Nelson, K. E., Metagenomic analysis of the human distal gut microbiome. *Science*, 312,5778(2006), pages 1355–1359.

HAQS07 Huson, D. H., Auch, A. F., Qi, J. and Schuster, S. C., MEGAN analysis of metagenomic data. *Genome Research*, 17,3(2007), pages 377–386. URL `http://genome.cshlp.org/content/17/3/377.abstract`.

HBB10 Horton, M., Bodenhausen, N. and Bergelson, J., MARTA: a suite of Java-based tools for assigning taxonomic status to DNA sequences. *Bioinformatics*, 26,4(2010), pages 568–569.

HDP+10 Hooper, S. D., Dalevi, D., Pati, A., Mavrommatis, K., Ivanova, N. and Kyrpides, N., Estimating DNA coverage and abundance in metagenomes using a gamma approximation. *Bioinformatics*, 26,3(2010), pages 295–301. URL `http://dblp.uni-trier.de/db/journals/bioinformatics/bioinformatics26.html#HooperDPMIK10`.

HGKM09 Haque, M. M., Ghosh, T. S., Komanduri, D. and Mande, S. S., SOrt-ITEMS: Sequence orthology based approach for improved taxonomic estimation of metagenomic sequences. *Bioinformatics*, 25,14(2009), pages 1722–1730.

HLP+13 Han, K., Li, Z.-f., Peng, R., Zhu, L.-p., Zhou, T., Wang, L.-g., Li, S.-g., Zhang, X.-b., Hu, W., Wu, Z.-h. et al., Extraordinary expansion of a Sorangium cellulosum genome from an alkaline milieu. *Scientific Reports*, 3,2101(2013), page 2101.

HMJ+12 Hunter, C. I., Mitchell, A., Jones, P., McAnulla, C., Pesseat, S., Scheremetjew, M. and Hunter, S., Metagenomic analysis: the challenge of the data bonanza. *Briefings in Bioinformatics*, 13,6(2012), pages 743–746.

HRB+98 Handelsman, J., Rondon, M. R., Brady, S. F., Clardy, J. and Goodman, R. M., Molecular biological access to the chemistry of unknown soil microbes: a new frontier for natural products. *Chemistry & Biology*, 5,10(1998), pages R245–R249.

HTAC+07 Handelsman, J., Tiedje, J., Alvarez-Cohen, L., Ashburner, M., Cann, I. K., DeLong, E., Doolittle, W., Fraser-Liggett, C., Godzik, A., Gordon, J. et al., Committee on Metagenomics: challenges and functional applications. *National Academy of Sciences, Washington*, 2007, pages 1–158.

JHA+15 Jones, M. B., Highlander, S. K., Anderson, E. L., Li, W., Dayrit, M., Klitgord, N., Fabani, M. M., Seguritan, V., Green, J., Pride, D. T. et al., Library preparation methodology can influence genomic and functional predictions in human microbiome research. *Proceedings of the National Academy of Sciences*, 112,45(2015), pages 14024–14029.

KDG+08 Krause, L., Diaz, N. N., Goesmann, A., Kelley, S., Nattkemper, T. W., Rohwer, F., Edwards, R. A. and Stoye, J., Phylogenetic classification of short environmental DNA fragments. *Nucleic Acids Research*, 36,7(2008), pages 2230–2239.

LGGP10 Liu, B., Gibbons, T., Ghodsi, M. and Pop, M., MetaPhyler: Taxonomic profiling for metagenomic sequences. *Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*. IEEE, 2010, pages 95–100.

LHJ+15 Land, M., Hauser, L., Jun, S.-R., Nookaew, I., Leuze, M. R., Ahn, T.-H., Karpinets, T., Lund, O., Kora, G., Wassenaar, T. et al., Insights from 20 years of bacterial genome sequencing. *Functional & Integrative Genomics*, 15,2(2015), pages 141–161.

LKZB13 Lo, C., Kim, S., Zakov, S. and Bafna, V., Evaluating genome architecture of a complex region via generalized bipartite matching. *BMC Bioinformatics*, 14,S-5(2013), page S13. URL `http://dx.doi.org/10.1186/1471-2105-14-S5-S13`.

LKZR13 Lindner, M. S., Kollock, M., Zickmann, F. and Renard, B. Y., Analyzing genome coverage profiles with applications to quality control in metagenomics. *Bioinformatics*, 29,10(2013), pages 1260–1267. URL `http://dblp.uni-trier.de/db/journals/bioinformatics/bioinformatics29.html#LindnerKZR13`.

LYY+11 Leung, H. C., Yiu, S.-M., Yang, B., Peng, Y., Wang, Y., Liu, Z., Chen, J., Qin, J., Li, R. and Chin, F. Y., A robust and accurate binning algorithm for metagenomic sequences with arbitrary species abundance ratio. *Bioinformatics*, 27,11(2011), pages 1489–1495.

MBCT15 Mäkinen, V., Belazzougui, D., Cunial, F. and Tomescu, A. I., *Genome-Scale Algorithm Design*. Cambridge University Press, 2015.

MGSM11 Mohammed, M. H., Ghosh, T. S., Singh, N. K. and Mande, S. S., SPHINX– an algorithm for taxonomic binning of metagenomic sequences. *Bioinformatics*, 27,1(2011), pages 22–30.

MHCM11 Miller, C. A., Hampton, O., Coarfa, C. and Milosavljevic, A., ReadDepth: A Parallel R Package for Detecting Copy Number Alterations

from Short Sequencing Reads. *PLoS ONE*, 6,1(2011), pages e16327+. URL `http://dx.doi.org/10.1371/journal.pone.0016327`.

MMG12 Mande, S. S., Mohammed, M. H. and Ghosh, T. S., Classification of metagenomic sequences: methods and challenges. *Briefings in Bioinformatics*, 13,6(2012), pages 669–681.

MMT+07 McHardy, A. C., Martín, H. G., Tsirigos, A., Hugenholtz, P. and Rigoutsos, I., Accurate phylogenetic classification of variable-length DNA fragments. *Nature Methods*, 4,1(2007), pages 63–72.

MPD+08 Meyer, F., Paarmann, D., D'Souza, M., Olson, R., Glass, E. M., Kubal, M., Paczian, T., Rodriguez, A., Stevens, R., Wilke, A. et al., The metagenomics RAST server–a public resource for the automatic phylogenetic and functional analysis of metagenomes. *BMC Bioinformatics*, 9,1(2008), page 386.

MTS+12 Morgan, X. C., Tickle, T. L., Sokol, H., Gevers, D., Devaney, K. L., Ward, D. V., Reyes, J. A., Shah, S. A., LeLeiko, N., Snapper, S. B. et al., Dysfunction of the intestinal microbiome in inflammatory bowel disease and treatment. *Genome Biology*, 13,9(2012), page R79.

NW10 Nelson, K. E. and White, B. A., Metagenomics and its applications to the study of the human microbiome. *Metagenomics: Theory, Methods and Applications*, 4,4(2010), pages 171–182.

Orl97 Orlin, J. B., A Polynomial Time Primal Network Simplex Algorithm for Minimum Cost Flows. *Mathematical Programming*, 78,2(1997), pages 109–129.

PBH+14 Pruitt, K. D., Brown, G. R., Hiatt, S. M., Thibaud-Nissen, F., Astashyn, A., Ermolaeva, O., Farrell, C. M., Hart, J., Landrum, M. J., McGarvey, K. M., Murphy, M. R., O'Leary, N. A., Pujar, S., Rajput, B., Rangwala, S. H., Riddick, L. D., Shkeda, A., Sun, H., Tamez, P., Tully, R. E., Wallin, C., Webb, D., Weber, J., Wu, W., DiCuccio, M., Kitts, P., Maglott, D. R., Murphy, T. D. and Ostell, J. M., RefSeq: an update on mammalian reference sequences. *Nucleic Acids Research*, 42,D1(2014), pages D756–D763. URL `http://nar.oxfordjournals.org/content/42/D1/D756.abstract`.

PSLO85 Pace, N. R., Stahl, D. A., Lane, D. J. and Olsen, G. J., Analyzing natural microbial populations by rRNA sequences. *ASM American Society for Microbiology News*, 51,1(1985), pages 4–12.

QLR+nt Qin, J., Li, R., Raes, J., Arumugam, M., Burgdorf, K. S., Manichanh, C., Nielsen, T., Pons, N., Levenez, F., Yamada, T., Mende, D. R., Li, J., Xu, J., Li, S., Li, D., Cao, J., Wang, B., Liang, H., Zheng, H., Xie, Y., Tap, J., Lepage, P., Bertalan, M., Batto, J.-M., Hansen, T., Le Paslier, D., Linneberg, A., Nielsen, H. B., Pelletier, E., Renault, P., Sicheritz-Ponten, T., Turner, K., Zhu, H., Yu, C., Li, S., Jian, M., Zhou, Y., Li, Y., Zhang, X., Li, S., Qin, N., Yang, H., Wang, J., Brunak, S., Dore, J., Guarner, F., Kristiansen, K., Pedersen, O., Parkhill, J., Weissenbach, J., Bork, P., Ehrlich, S. D. and Wang, J., A human gut microbial gene catalogue established by metagenomic sequencing. *Nature*, 464,7285(2010/03/04/print), pages 59–65. URL `http://dx.doi.org/10.1038/nature08821`.

RLL+03 Rocap, G., Larimer, F. W., Lamerdin, J., Malfatti, S., Chain, P., Ahlgren, N. A., Arellano, A., Coleman, M., Hauser, L., Hess, W. R., Johnson, Z. I., Land, M., Lindell, D., Post, A. F., Regala, W., Shah, M., Shaw, S. L., Steglich, C., Sullivan, M. B., Ting, C. S., Tolonen, A., Webb, E. A., Zinser, E. R. and Chisholm, S. W., Genome divergence in two Prochlorococcus ecotypes reflects oceanic niche differentiation. *Nature*, 424,6952(2003), pages 1042–1047. URL `http://dx.doi.org/10.1038/nature01947`.

RLY+11 Raes, J., Letunic, I., Yamada, T., Jensen, L. J. and Bork, P., Toward molecular trait-based ecology through integration of biogeochemical, geographical and metagenomic data. *Molecular Systems Biology*, 7,1(2011), page 473.

ROA+08 Richter, D. C., Ott, F., Auch, A. F., Schmid, R. and Huson, D. H., MetaSim—A Sequencing Simulator for Genomics and Metagenomics. *PLoS ONE*, 3,10(2008), page e3373. URL `http://dx.doi.org/10.1371%2Fjournal.pone.0003373`.

RRR11 Rosen, G. L., Reichenberger, E. R. and Rosenfeld, A. M., NBC: the Naive Bayes Classification tool webserver for taxonomic classification of metagenomic reads. *Bioinformatics*, 27,1(2011), pages 127–129.

RV11 Ribeca, P. and Valiente, G., Computational challenges of sequence classification in microbiomic data. *Briefings in Bioinformatics*, 12,6(2011), pages 614–625.

SBT+13 Segata, N., Boernigen, D., Tickle, T. L., Morgan, X. C., Garrett, W. S. and Huttenhower, C., Computational meta'omics for microbial community studies. *Molecular Systems Biology*, 9,1(2013), page 666.

SID+15 Schirmer, M., Ijaz, U. Z., D'Amore, R., Hall, N., Sloan, W. T. and Quince, C., Insight into biases and sequencing errors for amplicon sequencing with the Illumina MiSeq platform. *Nucleic Acids Research*, 43,6(2015), page e37.

SMZ+13 Sunagawa Shinichi, Mende Daniel R, Zeller Georg, Izquierdo-Carrasco Fernando, Berger Simon A, Kultima Jens Roat, Coelho Luis Pedro, Arumugam Manimozhiyan, Tap Julien, Nielsen Henrik Bjorn, Rasmussen Simon, Brunak Soren, Pedersen Oluf, Guarner Francisco, de Vos Willem M, Wang Jun, Li Junhua, Dore Joel, Ehrlich S Dusko, Stamatakis Alexandros and Bork Peer, Metagenomic species profiling using universal phylogenetic marker genes. *Nature Methods*, 10,12(2013), pages 1196–1199. URL `http://www.nature.com/nmeth/journal/v10/n12/abs/nmeth.2693.html\#supplementary-information`.

Ste56 Steinhaus, H., Sur la division des corps matériels en parties. *Bulletin of the Polish Academy of Sciences*, 1,12(1956), pages 801–804.

STM16 Sobih, A., Tomescu, A. I. and Mäkinen, V. *Research in Computational Molecular Biology: 20th Annual Conference, RECOMB 2016, Santa Monica, CA, USA, April 17-21, 2016, Proceedings*, chapter MetaFlow: Metagenomic Profiling Based on Whole-Genome Coverage Analysis with Min-Cost Flows, pages 111–121. Springer International Publishing, Cham, 2016. URL `http://dx.doi.org/10.1007/978-3-319-31957-5_8`.

SWB+12 Segata Nicola, Waldron Levi, Ballarini Annalisa, Narasimhan Vagheesh, Jousson Olivier and Huttenhower Curtis, Metagenomic microbial community profiling using unique clade-specific marker genes. *Nature Methods*, 9,8(2012), pages 811–814. URL `http://www.nature.com/nmeth/journal/v9/n8/abs/nmeth.2066.html\#supplementary-information`. 10.1038/nmeth.2066.

TCF+14 Tatusova, T., Ciufo, S., Fedorov, B., O'Neill, K. and Tolstoy, I., RefSeq microbial genomes database: new representation and annotation strategy. *Nucleic Acids Research*, 42,D1(2014), pages D553–D559.

TG08 Turnbaugh, P. J. and Gordon, J. I., An invitation to the marriage of metagenomics and metabolomics. *Cell*, 134,5(2008), pages 708–713.

THZ14 Tu, Q., He, Z. and Zhou, J., Strain/species identification in metagenomes using genome-specific markers. *Nucleic Acids Research*, 42,8(2014), pages e67–e67. URL `http://nar.oxfordjournals.org/content/early/2014/02/12/nar.gku138.abstract`.

TKRM13 Tomescu, A. I., Kuosmanen, A., Rizzi, R. and Mäkinen, V., A novel min-cost flow method for estimating transcript expression with RNA-Seq. *BMC Bioinformatics*, 14,Suppl 5(2013), page S15.

TWL+04 Teeling, H., Waldmann, J., Lombardot, T., Bauer, M. and Glöckner, F. O., TETRA: a web-service and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in DNA sequences. *BMC Bioinformatics*, 5,1(2004), page 163.

VAM+01 Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., Smith, H. O., Yandell, M., Evans, C. A., Holt, R. A. et al., The sequence of the human genome. *Science*, 291,5507(2001), pages 1304–1351.

WAC+08 Westbrooks, K., Astrovskaya, I., Campo, D., Khudyakov, Y., Berman, P. and Zelikovsky, A., HCV Quasispecies Assembly Using Network Flows. In *Bioinformatics Research and Applications*, Springer, 2008, pages 159–170.

WBPL91 Weisburg, W. G., Barns, S. M., Pelletier, D. A. and Lane, D. J., 16S ribosomal DNA amplification for phylogenetic study. *Journal of Bacteriology*, 173,2(1991), pages 697–703. URL `http://jb.asm.org/content/173/2/697.abstract`.

WLYC12 Wang, Y., Leung, H. C., Yiu, S.-M. and Chin, F. Y., MetaCluster 5.0: a two-round binning approach for metagenomic data for low-abundance species in a noisy sample. *Bioinformatics*, 28,18(2012), pages i356–i362.

WY11 Wu, Y.-W. and Ye, Y., A novel abundance-based algorithm for binning metagenomic sequences using l-tuples. *Journal of Computational Biology*, 18,3(2011), pages 523–534.

Appendix 1. MetaFlow in practice

Here we describe how MetaFlow can be used in practice. We explain what are the input files, output files, and the configuration parameters for the tool. The description is taken from our paper [STM16].

Input files

MetaFlow is a Linux-based C++ tool which runs from the command line. There are two input files for the tool:
A mapping file: This is the aligner's output file transformed into another format suitable for MetaFlow. We provide a python script which converts BLAST's output into our mapping file. If the user wishes to use another aligner, she/he should transform the aligner's output into our mapping file format.
A species database file: A file contains the name and the genome length for all the species in the database used by the aligner. We provide a default file created from NCBI bacterial database [PBH+14]. A tutorial that explains the format of these files and how to create them exists in our tool documentation.

Output files

MetaFlow generates different output files. All the files are in CSV format to make any further analysis easy. Table 6 lists the output files and their description.

MCF.config configuration file

We provide a set of configuration parameters which can be used to tune the behavior of our model for finding the approximated relative abundance, and to control the running time of the tool. The MCF.config is a configuration file that contains these parameters. The parameters are new line-separated, whereas the parameters and their values are tab-separated. Here, we list these parameters, their description and default values.

CHUNK_SIZE: Each genome is split into chunks of equal sizes, into which the reads are mapped based on their mapping position given by the aligner. If the chunk size is too large, e.g. 50,000, then the uniformity assumption in read

coverage is not fully exploited. On the other hand, if it is too short, e.g. 500, then the running time becomes unfeasible. The chunk size should be decided based on the average read length. For example, it can be selected to be ten times the average read length (default=2000).

ALPHA (α): The weight of the uniform distribution constraint in the optimization problem explained in the main paper. If α is too small, the aligner mapping score will dominate, and the uniform distribution constraint will be almost neglected. If α is too large, uniform distribution constraint will dominate, and the mapping score will be almost neglected ($\alpha \in [0, 1]$, default=0.9).

TRIMMING_PERCENTAGE (TR): As explained in the main paper, we iterate the minimum-cost flow algorithm, at each step updating the solution vector R^j of read mappings inside each genome to R^{j+1}, using the TR-trimmed mean read coverage of the chunks of each genome G^i, obtained from M^j. TR is thus the trimmed mean value ($0 \leq TR \leq 0.5$, default=0.2). A trimmed mean with value $TR = 0.2$ is calculated by trimming 40% of the chunks (the lowest 20% and highest 20% based on their coverage), and calculating the mean for the rest 60%. If $TR = 0$, then the required abundance per chunk in G^i will be the mean coverage for all chunks. If $TR = 0.5$, the required abundance per chunk will be the median coverage of the chunks.

In practice, it is not true that the read coverage of each base (in our case, of each chunk) is the same across each reference genome, but it follows a distribution. The are various proposals for this distribution, for example the negative binomial [MHCM11] or gamma [HDP+10], or mixtures of distributions [LKZR13]. In order to achieve a better approximation for the true distribution, we introduce two parameters for relaxing the objective function

$$\left| \frac{read\_mappings(i)}{s_i} - d_M(y_j^i) \right|$$

of the coverage-sensitive metagenomic mapping problem and allowing for a more practical formulation.

SPLIT_ARCS (SA): A boolean value (0 for false and 1 for true) allows for switching between a strict and a relaxed uniform distribution constraint. If $SA = 1$, the uniform distribution constraint is relaxed, allowing chunks to have coverage higher or lower than the required coverage based on the value of the parameter NUM_OF_READS_WITH_LOWER_COST. This is achieved by

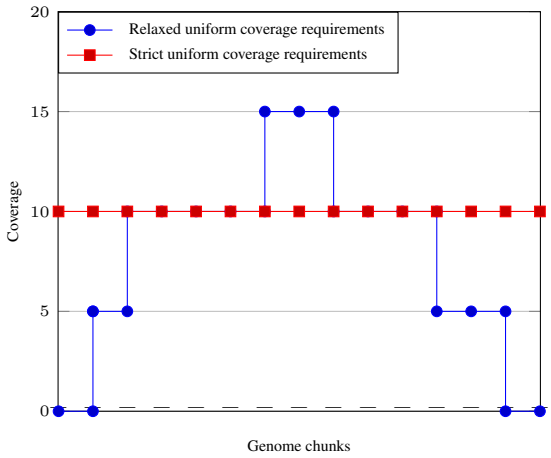


Figure 9: Comparison between the strict and relaxed uniform distribution requirments. The strict uniform distribution requirment (in red) requires all chunks to be covered with 10 reads. The relaxed uniform distribution requirement (in blue) also requires all chunks to be covered with 10 reads. However, the parameter NUM_OF_READS_WITH_LOWER_COST=5 allows for some chunks to have coverage 5, and some other chunks for have coverage up to 15. Also, the parameter REQUIRED_MAX_PER_OF_EMPTY_CHUNKS allows for having completely empty chunks. This allows for a better approximation for the true distribution of the reads.

penalizing these variations less than the default coverage penalty. If $SA = 0$ any variations from the required coverage will be penalized the default penalty, which makes the uniform distribution constraint strict (default=1).

NUM_OF_READS_WITH_LOWER_COST (N_{lc})**:** The number of reads per each chunk that will be penalized less in case of coverage lower or higher than the required one ($N_{lc} \in N^0$, default=5). Fig 9 gives an example of how these two parameters relax the uniform distribution constraint.

Deciding whether a species is an outlier or not is based on the absolute abundance of the species, and on the coverage of its chunks. For example, if the absolute abundance of a species is 0.1, then with high probability it is an outlier, and should be removed from the sample. We use the following parameters for deciding whether a species is an outlier or not. The user can choose to relax or tighten their values based on her knowledge about the input same.

REQUIRED_MIN_ABUNDANCE ($rel\_abundance_{min}$)**:** For any species, if the

absolute abundance is lower than $rel\_abundance_{min}$, then the species is considered an outlier and will be removed ($rel\_abundance_{min} \in [0, 1]$, default=0.3).

REQUIRED_AVERAGE_CHUNKS_COVERAGE ($Coverage_{avg}$): For any species, if the average coverage of chunks (number of reads/number of chunks) is lower than $Coverage_{avg}$, then the species is considered an outlier and will be removed ($Coverage_{avg} \in \mathbb{N}$, default=2).

REQUIRED_MAX_PER_OF_EMPTY_CHUNKS ($Chunks_{emp}$): For any species, if the ratio between number of chunks not covered by any read and the total number of chunks is more than $Chunks_{emp}$, then the species will be considered an outlier and will be removed ($Chunks_{emp} \in [0, 1]$, default=0.4).

The size of the flow network (number of nodes, and number of arcs) affects the running time for the MCF solver. The following parameters are used for controlling the number of arcs in a flow network.

MAX_SCORE_DIFFERENCE (S_{diff}): For each read, if the difference between the score of the best hit and any other hit is larger than S_{diff}, remove this hit ($S_{diff} \in \mathbb{N}$, default=0).

MAX_NUMBER_OF_ARCS: If the number of arcs in a flow network is larger than this value, randomly remove arcs with higher mapping cost until the number of arcs less than this (default=5,000,000).

As explained before, finding an approximation for the optimum abundance is achieved by running the minimum-cost flow algorithm iteratively. The following parameters are used for controlling the running time.

MAX_COST_DIFFERENCE (C_{diff}): For any two consecutive iterations itr_k, itr_{k+1} with solution values C_k, C_{k+1}, if $|C_k - C_{k+1}| \leq C_{diff}$, terminate the loop and return the solution of itr_{k+1} as the approximated abundance ($C_{diff} \in \mathbb{N}$, default=1).

MAX_READS_DIFFERENCE (R_{diff}): For any two consecutive iterations, if the difference between the total number of reads remained after each iteration is larger than R_{diff}, terminate the loop and return the current solution ($R_{diff} \in \mathbb{N}$, default=0).

File	Description
abundance.csv	The main output file. Contains the final estimation of the species richness and abundances.
dist.csv	Contains the final distribution of the reads over the genomes' chunks.
step0.abundance.csv step1.abundance.csv step2.abundance.csv step3.abundance.csv	Contains the estimation of the richness and the abundances before starting, and after stages 1, 2, and 3 respectively.
step0.dist.csv step1.dist.csv step2.dist.csv step3.dist.csv	Contains the distribution of the reads over the genomes' chunks before starting, and after stages 1, 2, and 3 respectively.
.log	The running log file.

Table 6: Application output files

MAX_NUMBER_OF_LOOPS (N_{loops}): The total number of iterations for finding the approximated abundance. If the number of iterations exceeded this value, terminate the loop and return the current solution as the approximated abundance ($N_{loops} \in \mathbb{N}^+$, default=10).

MAX_RUNNING_TIME: If the running time (in minutes) for finding the approximated abundance exceeded this value, terminate the loop and return the current solution as the approximated abundance (default=300).

MAX_NUMBER_OF_ARCS: If the number of arcs in a flow network larger than this, randomly remove arcs with higher score until the number of arcs less than this (default=5000000). This is for controlling the number or arcs in the flow networks.

Lemon library

Lemon library[DJK11] is an open source C++ library, which provides implementations for different data structures and algorithms. It provides an implementation of the network simplex algorithm [Orl97] which was used to solve our minimum cost flow problem.

Appendix 2. LC-Known and LC-Unknown results

The complete results for LC-Known and LC-Unknown experiments are shown in Table 7 and 8.

Sample ID	Tool	TP	FP	FN	Sensitivity	Precision	F-measure	L_1-norm
LC1	MetaFlow	15	0	0	1.000	1.000	**1.000**	0.72
	MetaPhlAn	15	1	0	1.000	0.938	0.968	5.01
	BLAST	15	0	0	1.000	1.000	**1.000**	**0.44**
	GSMer	7	31	8	0.467	0.184	0.264	N/A
	mOTU	15	1	0	1.000	0.938	0.968	7.43
LC2	MetaFlow	15	0	0	1.000	1.000	**1.000**	**0.66**
	MetaPhlAn	15	2	0	1.000	0.882	0.938	13.74
	BLAST	15	5	0	1.000	0.750	0.857	8.33
	GSMer	8	36	7	0.533	0.182	0.271	N/A
	mOTU	14	3	1	0.933	0.824	0.875	54.06
LC3	MetaFlow	14	1	1	0.933	0.933	0.933	2.12
	MetaPhlAn	15	1	0	1.000	0.938	**0.968**	37.2
	BLAST	15	2	0	1.000	0.882	0.938	**1.98**
	GSMer	9	45	6	0.600	0.167	0.261	N/A
	mOTU	14	1	1	0.933	0.933	0.933	5.61
LC4	MetaFlow	15	0	0	1.000	1.000	**1.000**	**1.71**
	MetaPhlAn	15	1	0	1.000	0.938	0.968	11.54
	BLAST	15	3	0	1.000	0.833	0.909	3.99
	GSMer	8	58	7	0.533	0.121	0.198	N/A
	mOTU	15	2	0	1.000	0.882	0.938	20.29
LC5	MetaFlow	15	0	0	1.000	1.000	**1.000**	**1.19**
	MetaPhlAn	15	1	0	1.000	0.938	0.968	23.04
	BLAST	15	4	0	1.000	0.789	0.882	12.2
	GSMer	9	5	6	0.600	0.643	0.621	N/A
	mOTU	15	1	0	1.000	0.938	0.968	35.22
LC6	MetaFlow	15	0	0	1.000	1.000	**1.000**	0.48
	MetaPhlAn	15	0	0	1.000	1.000	**1.000**	23.63
	BLAST	15	0	0	1.000	1.000	**1.000**	**0.13**
	GSMer	8	13	7	0.533	0.381	0.444	N/A
	mOTU	15	0	0	1.000	1.000	1.000	2.13
LC7	MetaFlow	15	0	0	1.000	1.000	**1.000**	**0.49**
	MetaPhlAn	15	2	0	1.000	0.882	0.938	18.31
	BLAST	15	1	0	1.000	0.938	0.968	0.54
	GSMer	8	52	7	0.533	0.133	0.213	N/A
	mOTU	14	0	1	0.933	1.000	0.966	11.84
LC8	MetaFlow	15	0	0	1.000	1.000	**1.000**	1.09
	MetaPhlAn	15	0	0	1.000	1.000	**1.000**	2.76
	BLAST	15	0	0	1.000	1.000	**1.000**	**0.7**
	GSMer	6	103	9	0.400	0.055	0.097	N/A
	mOTU	15	0	0	1.000	1.000	1.000	2.29
LC9	MetaFlow	15	0	0	1.000	1.000	**1.000**	0.49
	MetaPhlAn	15	1	0	1.000	0.938	0.968	3.2
	BLAST	15	0	0	1.000	1.000	**1.000**	**0.15**
	GSMer	13	20	2	0.867	0.394	0.542	N/A
	mOTU	14	1	1	0.933	0.933	0.933	27.46
LC10	MetaFlow	15	0	0	1.000	1.000	**1.000**	**0.88**
	MetaPhlAn	15	0	0	1.000	1.000	**1.000**	16.74
	BLAST	15	3	0	1.000	0.833	0.909	1.88
	GSMer	11	91	4	0.733	0.108	0.188	N/A
	mOTU	15	1	0	1.000	0.938	0.968	17.45
LC11	MetaFlow	15	1	0	1.000	0.938	**0.968**	**9.74**
	MetaPhlAn	15	1	0	1.000	0.938	**0.968**	56.09
	BLAST	15	7	0	1.000	0.682	0.811	17.48
	GSMer	11	70	4	0.733	0.136	0.229	N/A
	mOTU	13	2	2	0.867	0.867	0.867	56.61
LC12	MetaFlow	15	2	0	1.000	0.882	**0.938**	**13.9**
	MetaPhlAn	14	4	1	0.933	0.778	0.848	26.77
	BLAST	15	5	0	1.000	0.750	0.857	20.78
	GSMer	9	81	6	0.600	0.100	0.171	N/A
	mOTU	12	1	3	0.800	0.923	0.857	98.17
LC13	MetaFlow	15	3	0	1.000	0.833	0.909	**40.81**
	MetaPhlAn	14	2	1	0.933	0.875	0.903	56.72
	BLAST	15	7	0	1.000	0.682	0.811	41.04
	GSMer	8	62	7	0.533	0.114	0.188	N/A
	mOTU	14	1	1	0.933	0.933	**0.933**	54.81
LC14	MetaFlow	15	3	0	1.000	0.833	**0.909**	60.74
	MetaPhlAn	13	3	2	0.867	0.813	0.839	64.94
	BLAST	15	5	0	1.000	0.750	0.857	**60.4**
	GSMer	7	67	8	0.467	0.095	0.157	N/A
	mOTU	12	1	3	0.800	0.923	0.857	104.94
LC15	MetaFlow	15	2	0	1.000	0.882	**0.938**	21.19
	MetaPhlAn	15	2	0	1.000	0.882	**0.938**	36.74
	BLAST	15	3	0	1.000	0.833	0.909	**16.99**
	GSMer	8	102	7	0.533	0.073	0.128	N/A
	mOTU	10	1	5	0.667	0.909	0.769	46.43

Table 7: Detailed results on the LC-Known datasets. Sample descriptions are in Table 1. Best results are shown in bold. TP = true positives, FP = false positives, FN = false negatives.

Sample ID	Tool	TP	FP	FN	Sensitivity	Precision	F-measure	L_1-norm
LC1	MetaFlow	12	4	3	0.800	0.750	0.774	6.33
	MetaPhlAn	12	2	3	0.800	0.857	**0.828**	**5.51**
	BLAST	12	7	3	0.800	0.632	0.706	6.44
	GSMer	8	22	7	0.533	0.267	0.356	N/A
	mOTU	12	2	3	0.800	0.857	**0.828**	11.36
LC2	MetaFlow	12	1	3	0.800	0.923	**0.857**	**15.26**
	MetaPhlAn	12	5	3	0.800	0.706	0.750	38.01
	BLAST	12	1	3	0.800	0.923	**0.857**	19.44
	GSMer	5	28	10	0.333	0.152	0.208	N/A
	mOTU	12	1	3	0.800	0.923	**0.857**	29.06
LC3	MetaFlow	12	1	3	0.800	0.923	**0.857**	**3.37**
	MetaPhlAn	12	2	3	0.800	0.857	0.828	15.97
	BLAST	12	2	3	0.800	0.857	0.828	3.55
	GSMer	6	38	9	0.400	0.136	0.203	N/A
	mOTU	11	1	4	0.733	0.917	0.815	10.73
LC4	MetaFlow	12	0	3	0.800	1.000	**0.889**	**1.68**
	MetaPhlAn	12	1	3	0.800	0.923	0.857	12.19
	BLAST	12	2	3	0.800	0.857	0.828	3.90
	GSMer	4	57	11	0.267	0.066	0.105	N/A
	mOTU	12	1	3	0.800	0.923	0.857	7.35
LC5	MetaFlow	12	0	3	0.800	1.000	**0.889**	**1.18**
	MetaPhlAn	12	5	3	0.800	0.706	0.750	29.54
	BLAST	12	8	3	0.800	0.600	0.686	17.01
	GSMer	8	12	7	0.533	0.400	0.457	N/A
	mOTU	12	4	3	0.800	0.750	0.774	42.09
LC6	MetaFlow	12	0	3	0.800	1.000	**0.889**	**0.51**
	MetaPhlAn	12	2	3	0.800	0.857	0.828	27.65
	BLAST	12	1	3	0.800	0.923	0.857	3.81
	GSMer	8	1	7	0.533	0.889	0.667	N/A
	mOTU	12	1	3	0.800	0.923	0.857	22.56
LC7	MetaFlow	12	2	3	0.800	0.857	**0.828**	**20.24**
	MetaPhlAn	12	4	3	0.800	0.750	0.774	27.29
	BLAST	12	3	3	0.800	0.800	0.800	23.23
	GSMer	8	46	7	0.533	0.148	0.232	N/A
	mOTU	11	1	4	0.733	0.917	0.815	25.34
LC8	MetaFlow	12	0	3	0.800	1.000	**0.889**	**1.15**
	MetaPhlAn	12	2	3	0.800	0.857	0.828	7.28
	BLAST	12	1	3	0.800	0.923	0.857	4.95
	GSMer	7	121	8	0.467	0.055	0.098	N/A
	mOTU	12	1	3	0.800	0.923	0.857	26.80
LC9	MetaFlow	12	1	3	0.800	0.923	**0.857**	**7.36**
	MetaPhlAn	12	4	3	0.800	0.750	0.774	13.12
	BLAST	12	2	3	0.800	0.857	0.828	9.65
	GSMer	7	13	8	0.467	0.350	0.400	N/A
	mOTU	11	2	4	0.733	0.846	0.786	31.78
LC10	MetaFlow	12	5	3	0.800	0.706	0.750	**17.92**
	MetaPhlAn	12	3	3	0.800	0.800	**0.800**	22.51
	BLAST	12	11	3	0.800	0.522	0.632	19.86
	GSMer	6	116	9	0.400	0.049	0.088	N/A
	mOTU	12	3	3	0.800	0.800	0.800	31.94
LC11	MetaFlow	12	6	3	0.800	0.667	0.727	**37.97**
	MetaPhlAn	12	3	3	0.800	0.800	**0.800**	67.58
	BLAST	12	15	3	0.800	0.444	0.571	46.94
	GSMer	11	89	4	0.733	0.110	0.191	N/A
	mOTU	10	4	5	0.667	0.714	0.690	79.81
LC12	MetaFlow	12	2	3	0.800	0.857	**0.828**	**15.18**
	MetaPhlAn	11	8	4	0.733	0.579	0.647	31.62
	BLAST	12	8	3	0.800	0.600	0.686	25.77
	GSMer	7	108	8	0.467	0.061	0.108	N/A
	mOTU	9	4	6	0.600	0.692	0.643	113.43
LC13	MetaFlow	12	3	3	0.800	0.800	**0.800**	**53.04**
	MetaPhlAn	11	2	4	0.733	0.846	0.786	72.58
	BLAST	12	7	3	0.800	0.632	0.706	53.10
	GSMer	5	64	10	0.333	0.072	0.119	N/A
	mOTU	11	2	4	0.733	0.846	0.786	72.70
LC14	MetaFlow	12	3	3	0.800	0.800	**0.800**	**59.75**
	MetaPhlAn	10	7	5	0.667	0.588	0.625	66.82
	BLAST	12	6	3	0.800	0.667	0.727	63.36
	GSMer	7	82	8	0.467	0.079	0.135	N/A
	mOTU	9	3	6	0.600	0.750	0.667	110.04
LC15	MetaFlow	12	3	3	0.800	0.800	**0.800**	27.11
	MetaPhlAn	12	6	3	0.800	0.667	0.727	34.64
	BLAST	12	4	3	0.800	0.750	0.774	**21.09**
	GSMer	5	125	10	0.333	0.038	0.069	N/A
	mOTU	8	1	7	0.533	0.889	0.667	41.17

Table 8: Detailed results on the LC-Unknown datasets. Best results are shown in bold. TP = true positives, FP = false positives, FN = false negatives.