Clustering Metagenomic Sequences Using Canopies

Mohammad Arifur Rahman, Nathan LaPierre, Huzefa Rangwala and Daniel Barbara
Department of Computer Science
George Mason University
Fairfax, VA, United States
Email: mrahma23@gmu.edu, nlapier2@gmu.edu, rangwala@cs.gmu.edu and dbarbara@gmu.edu

***Abstract*—Metagenomics is the study of genome sequences from samples which are retrieved directly from environment hosting multiple organisms that co-exist as communities within ecological regions. Metagenomics study has created the necessity and challenges of developing computational tools for the quantification of abundance, diversity, role of different species within communities, phenotype inferences etc. Enormous amount of data is created during metagenome sequencing process. Algorithms have been developed to cluster similar metagenome sequences. We have developed an approach for clustering metagenomic data that uses Canopy Clustering with Locality Sensitive Hashing distance approximation to make clustering process in metagenomic data faster. Canopy Clustering works as a preprocessing step to reduce pairwise distance calculation and enables efficient parallel processing with subsequent expensive cluster methods while LSH provides fast distance approximation and reduces data dimension. We tested our framework with three popular clustering mechanisms in literature on three synthetic and three real world large scale metagenome datasets and observed that our proposed approach can reduce runtime while providing similar in some cases better outcomes.**

***Keywords*-Clustering, Canopy, LSH, Metagenome**

I. Introduction

Metagenomics is the study of genetic material recovered directly from samples that comprise organisms as co-existing communities. These samples can be taken from sea, soil and human body [1][2]. Metagenomics has enabled scientists to study all of the genomes in a community as a whole. Analysis of microbial community through metagenome data can reveal interesting relationships between microbial community and the host which in turn can lead to further investigation. For example analyzing metagenomic data from human gut microbiome provides an understanding of the role played by microbes with regards to human health and disease [3].

Instead of producing whole genomic sequences of the members of communities in host samples, latest sequencing technologies produce short contiguous subsequences called *reads* from random positions of actual whole genome. These reads from different organisms are commingled together posing fundamental challenge to further analysis of the data. Combining the reads of different organisms based on overlapping yet discriminating information from organism specific genome sequences is known as sequence assembly [4]. Many Sequence assemblers require reference genome sequence and the process of assembly is exceedingly complex, challenging and time consuming [5]. For this reason clustering metagenome data for identification of Operational Taxonomic Units (OTU) from 16s rRNA genes has become popular recently. 16S rRNA gene sequencing has been widely used for the analysis of genetic diversity within complex microbial communities. 16S sequences are marker genes, which exists in most microbial species but have variations in their sequences that allow them to be separated into different taxonomic groups [6]. OTU is used to classify groups of closely related individuals from similar or different taxonomic levels [7]. It is the most commonly used microbial diversity unit, especially when analyzing the small subunit 16S or 18S rRNA marker gene microbial datasets [8]. Sequences can be clustered according to their similarity to each other. Microbial OTUs are generally ecologically consistent across the hosts regardless of OTU clustering approaches [9].

Clustering approaches have been developed and used for the rapid analysis of large sets of whole and targeted 16S rRNA metagenomic sequences which are discussed in Section II - Literature Review. Analysis of microbiome datasets typically begins by clustering raw sequence reads and creating potential OTUs. Proper clustering of sequence reads assists in the metagenome assembly problem, allows computation of different species diversity metrics and allows further analysis with phenotype inferences. In this study we have proposed and evaluated a pre-clustering technique based on Canopy cluster method and Locality Sensitive Hashing. Our proposed framework can reduce pairwise comparison between sequences for similarity measure in large scale metagenome datasets by partitioning the dataset with fast LSH based approximation. These initial partitions can be considered independent of each other. More accurate and expensive clustering methods can be deployed for each partition in parallel utilizing multi-core architechture of modern CPUs. Only the sequences inside a canopy will be considered for further sub-clustering not the sequences outside that canopy. This characteristic of Canopy clustering reduces expensive pairwise comparison significantly.

II. Literature Review

Some of the popular and early clustering methods for metagenomic clustering are CD-HIT [10], DOTUR [11], MOTHUR [12] and ESPRIT [13]. Algorithm used in CD-HIT is a greedy incremental clustering algorithm and performs ordering on sequences based on sequence length. Subsequent comparisons have to fall above the threshold for similarity with all sequences in a cluster. DOTUR, MOTHUR and ESPRIT exclusively use pairwise distance matrix as input and then perform hierarchical clustering clustering on 16S sequences. MOTHUR and DOTUR utilize global alignment score between all pairs of sequences for pairwise distances computation.

Hash based clustering mechanisms have been introduced for metagenome data. MC-LSH [14] utilizes an efficient locality sensitive based hashing function to approximate the pairwise sequence similarity. Similarity among sequences is computed based on randomly chosen indices that essentially compresses the input sequence. MC-MinH [15] algorithm uses the min-wise [16] hashing approach, along with a greedy clustering algorithm to group 16S and whole metagenomic sequences. Mash [17], uses MinHash locality sensitive hashing to reduce large sequences to a representative sketch and rapidly estimate pairwise distances. Other methods for clustering whole metagenome sequence reads include TOSS [18], AbundanceBin [19], and CompostBin [20]. All unique k-mers are first clustered in TOSS then clusters are merged based on k-mer repetition. In AbundanceBin sequencing reads are modeled as mixture of Poisson distributions. Then Maximization (EM) algorithm is used to infer model parameters and final clustering. Principal component analysis is used in CompostBin to project the data into lower dimensional space and then uses the normalized cut clustering [21] algorithm to group sequences into taxa specific clusters. CompostBin also uses phylogenetic markers to assign clusters.

UCLUST [22] follows a greedy process and creates *seeds* of sequences which generate clusters based on percent identity. In last two years, two relatively new clustering methods have been introduced Swarm [23][24] and SUMACLUST [25]. These methods are some of the currently popular, relatively faster [26] and open source softwares for clustering in Metagenome data. SWARM uses exhaustive single-linkage clustering based on optimal sequence alignment. SUMACLUST follows a similar approach as UCLUST. Abundance-ordered list of input sequences are compared against the representative set of already-chosen sequences in SUMACLUST. In this study we have introduced a very fast initial partitioning of large scale metagenomic datasets with Canopy Clustering [27] and Locality Sensitive Hashing [28][29][30]. This way efficient and scalable parallelism in OTU-clustering can be achieved that is capable of making existing OTU-clustering method multiple times faster by taking advantage of modern multicore CPU architectures. We have used UCLUST, SWARM and SUMACLUST with and without our proposed framework and brief descriptions of these methods are provided in Section III-D.

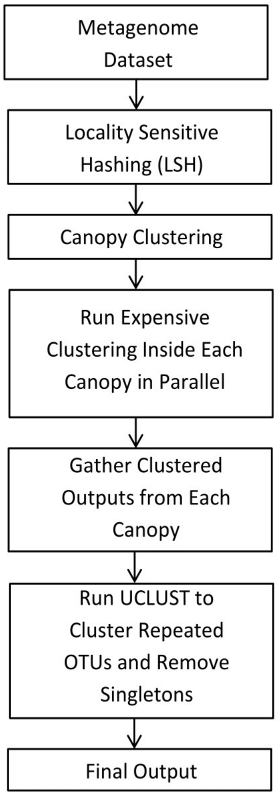

Metagenome Dataset
Locality Sensitive Hashing (LSH)
Canopy Clustering
Run Expensive Clustering Inside Each Canopy in Parallel
Gather Clustered Outputs from Each Canopy
Run UCLUST to Cluster Repeated OTUs and Remove Singletons
Final Output

Figure 1. Overview of our proposed Canopy Clustering approach on Metagenome Data

III. Methods

A. *Overview*

In this section we provide an overview of our proposed framework. Figure 1 represents an abstract overview of our proposed approach.

Figure 2 shows a more detailed overview of our proposed Canopy clustering approach for metagenomic data. After computing the kmers and LSH codes our proposed method will start canopy clustering based on the hamming distances between the binary representations of data points created by LSH. Once all reads in datasets are assigned to canopies, our method will create multiple partitions from original dataset based on canopy membership information. Finally each smaller partitions representing a canopy will be provided as input to more accurate sequence clustering methods in parallel. In this study UCLUST, SUMACLUST, SWARM and Minimum Spanning Tree (MST) based clustering will be used as accurate clustering methods for canopies which are briefly discussed in Section III-D.

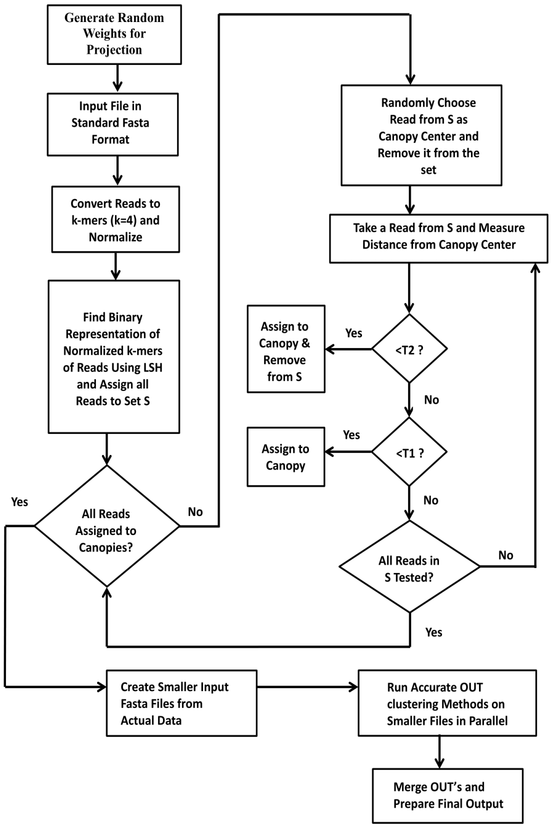

Generate Random Weights for Projection
Input File in Standard Fasta Format
Convert Reads to k-mers (k=4) and Normalize
Find Binary Representation of Normalized k-mers of Reads Using LSH and Assign all Reads to Set S
All Reads Assigned to Canopies?
Yes
No
Randomly Choose Read from S as Canopy Center and Remove it from the set
Take a Read from S and Measure Distance from Canopy Center
<T2 ?
Yes
Assign to Canopy & Remove from S
No
<T1 ?
Yes
Assign to Canopy
No
All Reads in S Tested?
No
Yes
Create Smaller Input Fasta Files from Actual Data
Run Accurate OUT clustering Methods on Smaller Files in Parallel
Merge OUT's and Prepare Final Output

Figure 2. Workflow of Canopy Clustering for Large Scale Metagenome Data

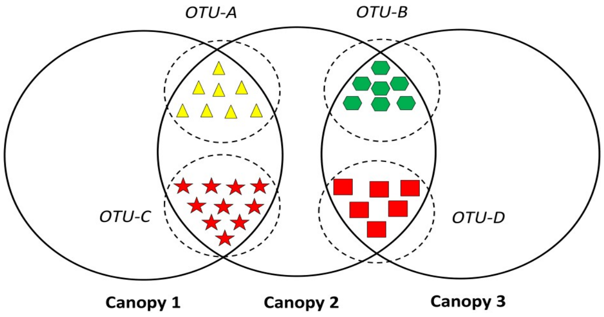


Figure 3. Same OTUs Occurring in Multiple Canopies

The final step of our proposed framework is to merge the clustered outputs from each canopy provided by more accurate and expensive clustering methods. There is a potential concern in this scenario. According to Canopy cluster algorithm a single data point may belong to multiple canopies as long as the soft threshold is met. As a result same or very similar OTUs will be generated from multiple canopies. This scenario is depicted in Figure 3. The dashed circles titled OTU-A, B, C and D represent *potential OTUs*. Soild outer circles represent canopies titled Canopy-1, 2 and 3. OTU-A and C are shared by Canopy-1 and 2. OTU-B and D are shared by Canopy-2 and 3. The smaller triangles, stars, hexagons and rectangles represent sequence reads in metagenome data. Because of this repetitions another final clustering on resulting OTUs is required which will remove repetitions in OTU. We are proposing two solutions to this scenario.

1) Merging with Greedy Clustering Algorithm: We have chosen UCLUST for merging step since UCLUST is one of the fastest greedy sequence clustering methods. We chose 90% identity threshold for UCLUST in this step. Other sequence clustering algorithm can also be applied in this step. Since number of sequences is already reduced by Canopy and accurate clustering, this merging step is very fast. Greedy clustering methods like UCLUST always converges to local minima. Thus this approach does not require number of OTUs as prior.

2) Merging with Minimum Spanning Tree: We developed a Minimum Spanning Tree and Locality Sensitive Hashing (LSH) distance based metagenome sequence clustering method to be used in merging step as well as inside Canopies for accurate clustering. A brief description of MST based clustering is given in Section III-D. Unlike merging with greedy clustering methods, this approach requires number of edges to remove as prior which indicates the number of OTUs.

In next sections we provide a brief description of Canopy Clustering, Locality Sensitive Hashing, their potential in metagenome data clustering, sub-clustering methods used in this study and our motivation for using them.

B. Canopy Clustering

Canopy Clustering [27] is an efficient approximate clustering algorithm often used as preprocessing step for other accurate and expensive clustering methods like the K-means algorithm or the Hierarchical clustering algorithm. It is intended to speed up clustering operations on large data sets, where standard clustering algorithms may be impractical in terms of run time and memory consumptions due to the large number of pairwise distance calculations. For a dataset with N instances, worst case calculations without canopy clustering is N^2. After canopy clustering if the number of canopies is k then worst case calculations with canopy clustering is $\sum_{i=1}^{k}(c_i)^2$ where c_i is the number of instances in ith canopy. The total difference g in pairwise distance calculations:

$$g = N^2 - \sum_{i=1}^{k}(c_i)^2 \qquad (1)$$

Canopy clustering uses two basic distance thresholds namely *soft* threshold $T1$ and *tight* threshold $T2$. If data point p_1 is within the soft distance threshold $T1$ with another data point p_2 then p_1 will reside in same canopy as p_2 but p_1 may belong to other canopies too assuming that it has only met soft threshold and best match is yet to be found. Thus one data point may belong to multiple canopies in Canopy clustering. On the other hand if data point p_1 is within the tight distance threshold $T2$ with another data point p_2 then canopy clustering assigns p_1 to the same canopy as p_2 and stops assigning p_1 to any other canopy assuming that tight threshold has been met and best canopy assignment for p_1 has been found. In this case p_1 will not be repeated in other canopies. Canopy centroids are selected randomly. Then all other data points are tested for canopy assignment based on $T1$ and $T2$. New canopies are selected until all data points are assigned to canopies. A fast approximation based on Locality Sensitive Hashing (LSH) will be used for Canopy membership identification.

C. *Locality Sensitive Hashing*

A Locality Sensitive Hashing (LSH) [28][29][30] is a distribution on a family $\mathcal{F}$ is defined for a metric space $\mathcal{M} = (M, d)$ a threshold $R > 0$ and an approximation factor $c > 1$. This family $\mathcal{F}$ is a family of functions $h : \mathcal{M} \rightarrow S$ which map elements from the metric space to a bucket $s \in S$. The LSH family either satisfies $h(p) = h(q)$ with probability at least P_1 if $d(p, q) \leq R$ or satisfies $h(p) = h(q)$ with probability at most P_2 if $d(p, q) \geq R$ for any two points $p, q \in \mathcal{M}$, using a function $h \in \mathcal{F}$ which is chosen randomly. A family is interesting when $P_1 > P_2$. Such a family $\mathcal{F}$ is called (R, cR, P_1, P_2)-sensitive. LSH has been used for fast comparison between points in very high dimensional space [30]. To preserve approximate cosine distance, data points are often randomly projected to low dimensional bit signatures [34].

In this study we have chosen a random projection based hashing function that projects a d dimensional data point into a n dimensional bit representation where $n < d$. The intuition behind choosing $n < d$ is to reduce dimension of data. Given a random projection v' of size $1 \times d$ and a data vector a of size $1 \times d$, the dot product between them provides a scalar value. Sign of this scalar value represents which side of the random hyperplane does the data point exist. This information can be represented with a bit. This way a single data point can be represented with limited number of bits. Hamming distance between these binary representations indicates *disagreements* between two data points.

D. *Sub-Clustering Inside Canopies*

We have used three recent and popular sequence clustering methods as the expensive clustering measure inside canopies in this study. UCLUST [22], SUMACLUST [25] and SWARM [23] were used for sub-clustering canopies.

1) UCLUST: It is a closed-source software and provides very limited documentations[1]. The 32 bit version of UCLUST executable is available for academic usage. But the 64-bit versions which is necessary to handle large datasets, require expensive license. UCLUST follows a greedy process and creates *seeds* of sequences which generate clusters based on percent identity.

2) SUMACLUST: An open-source software. It follows similar approach as UCLUST. Based on greedy strategy SUMACLUST incrementally constructs clusters by comparing an abundance-ordered list of input sequences against the representative set of already-chosen sequences. Initially this list is empty.

3) SWARM: SWARM uses exhaustive single-linkage clustering based on optimal sequence alignment. Sequences that are less than a certain distance from any other other sequence in the cluster are clustered together. SWARM attempts to reduce the impact of clustering parameters on the resulting OTUs by avoiding arbitrary global clustering thresholds and input sequence ordering dependence. At first SWARM builds an initial set of OTUs is constructed by iteratively agglomerating similar amplicons. Then amplicon abundance values are used to reveal OTUs internal structures and to break them into sub-OTUs.

4) Minimum Spanning Tree (MST): To evaluate Canopy as a standalone clustering method along with UCLUST, SUMACLUST and SWARM, we used Minimum Spanning Tree (MST) based clustering method as an accurate and expensive clustering inside canopies. Minimum Spanning Tree (MST) produces a connected spanning tree of undirected graph. It connects all vertices together such that the total weight of edges remains minimum. This property has been used for clustering [31][32][33]. We represented edges with Locality Sensitive Hashing based distances. Then edges with highest distances were removed from MST to generate OTUs.

In the following Sections UCLUST, SUMACLUST, SWARM and MST with Canopy clustering approach are represented as CC$_{UCLUST}$, CC$_{SUMACLUST}$, CC$_{SWARM}$ and CC$_{MST}$ respectively where the term CC stands for Canopy Clustering.

IV. Experimental Evaluation

A. *Dataset Description*

To evaluate the performance of our developed approach we have used previously published synthetic and real world sequence benchmarks. Specifically, we used three synthetic 16S rRNA gene mock community datasets (Bokulich$_2$, Bokulich$_3$, and Bokulich$_6$) from Bokulich et. al. [35] and three real data sets: a 16S rRNA gene soil data set (Canadian Soil) [36], a 16S rRNA gene human data set (Body Sites) [37] and 18S rRNA gene soil data set (Global Soil) [38]. Key

[1] http://www.drive5.com/uclust/uclust_userguide_1_1_579.pdf

Table I
DATASET STATISTICS

Datasets	Type	Reference	# of Reads	# of Samples	Read Length	Platform
Bokulich$_2$	Mock	[35]	6,938,836	4	189–251	HiSeq
Bokulich$_3$	Mock	[35]	3,594,237	4	114–151	HiSeq
Bokulich$_6$	Mock	[35]	250,903	1	114–150	HiSeq
Canadian Soil	Genuine	[36]	2,966,053	13	76–10	HiSeq
Body Sites	Genuine	[37]	886,630	602	117–351	GS FLX
Global Soil	Genuine	[38]	9,252,764	57	119–151	HiSeq

statistics and relevant information regarding these datasets are presented in Table I.

Synthetic Datasets:

1) Bokulich$_2$: This dataset was prepared using the Illumina TruSeq v2 paired–end library preparation kit. It is a simulated 16S rRNA gene microbial community data set. This dataset contains 19 taxonomic Families, 19 Genera, 22 Species and 22 Strains in total. This dataset can be found in the QIIME database (identifier 1685).

2) Bokulich$_3$: Similar to Bokulich$_2$ except that it was prepared with the TruSeq v1 paired-end library kit at Illumina Cambridge and is also available in the QIIME database (identifier 1686).

3) Bokulich$_6$: This 16S rRNA dataset was sequenced at Washington University School of Medicine and contains evenly distributed microbial communities. This dataset contains 13 taxonomic Families, 23 Genera, 44 Species and 48 Strains in total. All these datasets from Bokulich et al.[35] are available at QIIME database[2] under their respective ID's. Since, these are simulated datasets the taxonomic profile of microbial organisms within them are known.

Real World Datasets:

4) Canadian Soil: The Canadian Soil dataset[3] contains genomic data of soil spanning from Arctic Tundra to Agricultural soil suitable for different agricultural products.

5) Body Sites: This dataset contains composition of bacterial communities from up to 27 different body sites in healthy adults. A collection of 602 samples acquired from different body sites of human subjects are provided with meta-data.

6) Global Soil: The global soil data was taken from Ramirez et al. [38] which is a study of the below-ground diversity in New York City's Central Park.

All of these datasets have been used in previous studies [35],[39]. Any analysis on these datasets requires appropriate preprocessing which can significantly change the results of clustering and taxonomic classification based on them [35]. The performance of different open source sequence clustering methods were assessed and compared in a study by Kopylova et. al. [39] using these specific datasets. As such, in this paper we use the same benchmarks as done in the prior study and publicly available [4].

B. Evaluation Metrics

We assess the performance of our proposed approach using the following types of commonly used metrics: (i) that are used for the assessment of biodiversity within metagenomic samples, (ii) output of clustering algorithms and (iii) computational run time.

1) Faiths phylogenetic diversity metric (PD): Faiths phylogenetic diversity [40] is based on the Phylogenetic tree. It combines all the branch lengths of the tree as a measure of diversity. So, if a new OTU (cluster) is found and it is closely related to another OTU in the sample, it will contribute to a small increase to the PD score. However, if a new OTU is found and is from a different lineage than anything else in the sample, it will contribute to a large increase in the PD score.

2) Shannon Entropy: Shannon-Wiener diversity index is defined as:

$$H = -\sum_{i=1}^{s} (p_i \log_2 p_i) \tag{2}$$

where s is the number of OTUs and p_i is the proportion of the community represented by OTU i. Since by definition p_i will be between zero and one, the natural log makes all of the terms of the summation negative, which is why we take the inverse of the sum. The Shannon index increases as both the richness and evenness of the community increase. The fact that the index incorporates both components of biodiversity can be seen as both a strength and a weakness. It is a strength because it provides a simple summary, but it is a weakness because it makes it difficult to compare communities that differ greatly in richness.

3) Simpson's Index: Simpsons index is defined as $1 - dominance$ or

$$1 - \sum p_i^2 \tag{3}$$

where where p_i is the proportion of the community represented by OTU i. Simpsons index is based on the probability of any two individuals drawn at random from an infinitely large community belonging to the same species. It measures *evenness* of the community from 0 to 1. Higher value of this index refers higher similarity and relatively lower diversity of microorganisms.

[2] http://qiime.org/home_static/dataFiles.html

[3] http://www.cm2bl.org/

[4] https://github.com/ekopylova/otu-clustering

4) F-measure: In case of synthetic datasets, false-positive (FP; taxonomy/OTU string exists in observed but not expected), false-negative (FN; taxonomy/OTU string exists in expected but not observed), and true-positive (TP; taxonomy/OTU string exists in both observed and expected) measures were computed from cluster output and the ground truth which is the expected taxonomic composition. The following definitions were used:

$$precision = \frac{TP}{(TP + FP)} \tag{4}$$

$$recall = \frac{TP}{(TP + FN)} \tag{5}$$

$$FScore = \frac{2 \times precision \times recall}{(precision + recall)} \tag{6}$$

5) Pearson Coefficient Correlation (ρ-value): After getting Operational Taxonomic Units (OTU) from a clustering method we created the taxonomic profile at Genus level. Pearsons correlation coefficient was computed to measure the relatedness of taxonomic assignment between a pair of tools. Values can range between -1 and 1, with -1 indicating a negative correlation, 0 indicating no correlation, and 1 indicating a positive correlation or strong relationship.

C. Experimental Details

For kmers, the value of parameter K was set to 4. There are 4 possible nucleotide representatives (A,C,G and T) and the total number of features became $4^4 = 256$. This way we were able to convert the string representation into numeric. For Locality Sensitive Hashing the number of Hyperplanes (parameter d) was set to 150. We wanted to reduce the number of features while maintaining a relatively accurate distance approximation. So value of d was chosen to be less than 256. For performance analysis and comparison we have chosen $T1 = 0.45$ and $T2 = 0.34$.

We performed all the experiments on computers with Intel 5th generation Core i7 2.70GHz 64bit processor with 8 core CPUs and 12GB memory. For implementation we used Python 2.7.12 and QIIME [41] version 1.9.0 - a popular open source Bioinformatics pipeline that combines many metagenome clustering methods including the ones which has been used in this work. One important aspect our proposed approach is parallelism. Each canopy can be clustered in parallel. To take full advantage of today's multicore CPU based computing systems we utilized Python's *multiprocessing* module instead of *threading* module. According to Python's documentation[5] *multiprocessing* module allows the programmer to fully leverage multiple processors on a given machine by spawning subprocesses instead of threads. After receiving output from all canopy clustering we merged the outputs to make a single clustering results. Some performance metric used in this study like Faiths Phylogenetic Diversity (PD) metric require sequence alignment. We used PyNast[6] [42] open source sequence aligner for aligning clustered output.

V. Results Discussion

A. Clustering Performance Comparison

Table II shows the performance of UCLUST [22], SUMACLUST [25] and SWARM [23] and their corresponding versions with our proposed approach. Table II compares F-scores and Pearson Correlation Coefficient (ρ). F-scores are only available for synthetic benchmarks since taxanomic profile for them is known as ground truth. But F-scores are not available for real benchmarks as no ground truth is available for them. Correlation values were generated based on taxonomic profiles at Genus level generated from outputs from clustering methods. From Table II we can see that F-scores obtained from a clustering method and its corresponding Canopy clustering version are very closer, in some cases better. We got higher F-score for Bokulich$_2$ benchmark from CC$_{SUMACLUST}$. CC$_{UCLUST}$ and CC$_{SWARM}$ provided same F-scores as their respective naive versions for Bokulich$_2$. For Bokulich$_3$ benchmark we observed higher F-scores for all LSH based Canopy clustering methods. Finally for Bokulich$_6$ benchmark F-scores of UCLUST and SWARM were improved comparing to respective naive versions.

For all benchmarks we got very strong correlations between taxonomic profiles at genus level from UCLUST, SUMACLUST, SWARM and their respective LSH based Canopy counterparts. The highest correlation was observed for Bokulich$_6$ benchmark between UCLUST and CC$_{UCLUST}$ with 0.9831 and the lowest was observed for Canadian Soil metagenome data between SUMACLUST and CC$_{SUMACLUST}$ with 0.7581. From observed F-scores and correlation values we can say that running a sequence clustering method with our proposed LSH based Canopy cluster framework can bring similar or better results. Next we will discuss about Runtime Comparisons. CC$_{MST}$ provided relatively similar F-scores and high ρ values indicating strong correlations of taxonomic outcomes with other methods.

B. Runtime Comparison

Table III shows the runtime in minutes of UCLUST, SUMACLUST, SWARM and their respective versions with our proposed Canopy clustering pipeline. Time information in Table III was recorded while populating performance metric values for Table II for the parameter setting mentioned in Experimental Details section. Improving runtime while maintaining similar or better clustering results was one of the major motivations of our proposed approach. We can see from Table III that CC$_{UCLUST}$ outperforms

[5] https://docs.python.org/2/library/multiprocessing.html

[6] http://biocore.github.io/pynast/

UCLUST for all datasets. The highest speed up for UCLUST was observed for Bokulich$_2$ dataset where UCLUST was 2.10 times faster with LSH based Canopy than the naive version. For SUMACLUST we can see from Table III that CC$_{SUMACLUST}$ outperforms naive SUMACLUST in most cases. Major improvement in runtime was observed in large scale real world metagenome datasets specially Canadian Soil and Global Soil datasets. The highest speed-up for SUMACLUST with Canopy clustering was observed to be 7.70x for Global Soil dataset.

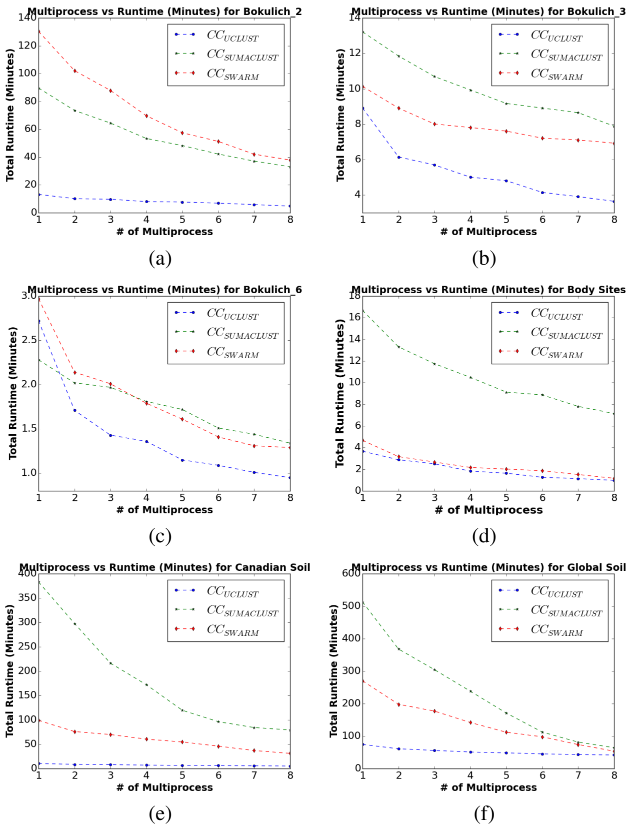
We can see a similar footprints for CC$_{SWARM}$ from Table III. The highest reduction in runtime for SWARM with Canopy clustering was observed to be 4.79x for Global Soil dataset. From these observations we can conclude that LSH based Canopy clustering not only reduces the runtime of an expensive clustering method but also scales well and performs better for larger benchmarks comparing to the smaller ones.

C. Biodiversity Comparison

Clustering sequences in metagenome data will provide OTUs that represent biodiversity contained in the samples from which data is collected. Hence we also need to compare the biodiversity represented by OTUs from naive methods and their respective Canopy clustering variations used in this study. Table IV shows Faiths phylogenetic diversity metric (PD), Shannon and Simpson index after clustering with different methods. These metrics are some of the popular Alpha Diversity metrics meaning that they measure species diversity in sites or habitats at a local scale. These metric values are generated per sample basis. Table IV shows ranges of metric values in $[minimum - maximum]$ format provided by methods with and without Canopy clustering. Any sample of a dataset will take value from this range. These ranges may not be same since OTUs vary over clustering methods. But they should be similar. We can see from Table IV that Canopy clustering based methods produce similar ranges of values as their naive counterparts. No significant changes in diversity metric values were observed which indicates that Canopy based approaches reproduce similar biodiversity while reducing runtime.

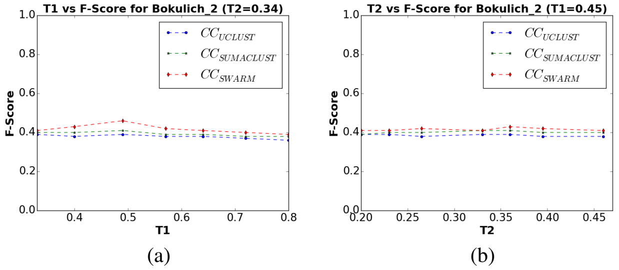
D. Effect of Varying Number of Multi-processes

Figure 4a-4f shows how the number of multiprocess can affect runtime of CC$_{UCLUST}$, CC$_{SUMACLUST}$ and CC$_{SWARM}$ on six benchmarks used in this study. We observed that increasing the number of multiprocess can reduce total runtime. The steepest curve showing reduction in runtime can be found in Figure 4e for CC$_{SUMACLUST}$ on Canadian Soil dataset. For single process version CC$_{SUMACLUST}$ is slower in most cases except Bokulich$_2$ dataset where it's runtime is better comparing to CC$_{SWARM}$. For Body Site dataset (Figure 4d) the CC$_{UCLUST}$ and CC$_{SWARM}$ have closer runtime with respect to increasing number of multiprocess. Figure 4f shows reduction in runtime for the largest dataset used in this study namely Global Soil with more than nine million reads.



(a) (b) (c) (d) (e) (f)

Figure 4. Runtime comparison of Canopy clustering framework with UCLUST, SUMACLUST and SWARM on Bokulich$_2$, Bokulich$_3$, Bokulich$_6$, Body Sites, Canadian Soil and GLobal Soil datasets.



(a) (b)

Figure 5. 5a and 5b show effect of varying $T1$ and $T2$ on F-scores for the largest synthetic dataset Bokulich$_2$

E. Effect of Varying T1 and T2 Parameter of Canopy Clustering

Figure 5a-5b shows effect of varying $T1$ ($T2$ fixed at 0.34) and $T2$ ($T1$ fixed at 0.45) on F-scores for the largest synthetic dataset Bokulich$_2$. Reducing $T1$ means comparatively *tighter* soft-threshold which will reduce repetitions of instances in multiple canopies. For a fixed $T2$ this

Table II
PERFORMANCE COMPARISON [F-SCORE AND PEARSON CORRELATION COEFFICIENT (ρ)]

Methods	**Comparison Metric**	**Datasets**					
		Synthetic			*Real World*		
		Bokulich$_2$	Bokulich$_3$	Bokulich$_6$	Body Sites	Canadian Soil	Global Soil
UCLUST	F-Measure	0.39	0.40	0.51	N/A	N/A	N/A
CC$_{UCLUST}$	F-Measure	***0.39***	***0.41***	***0.52***	N/A	N/A	N/A
	Correlation (ρ) with Respect to UCLUST	***0.9831***	***0.9753***	***0.9831***	***0.9682***	***0.8419***	***0.9824***
SUMACLUST	F-Measure	0.40	0.41	0.51	N/A	N/A	N/A
CC$_{SUMACLUST}$	F-Measure	***0.41***	***0.42***	***0.51***	N/A	N/A	N/A
	Correlation (ρ) with Respect to SUMACLUST	***0.9709***	***0.9813***	***0.9538***	***0.9518***	***0.7643***	***0.8714***
SWARM	F-Measure	0.46	0.48	0.55	N/A	N/A	N/A
CC$_{SWARM}$	F-Measure	***0.46***	***0.49***	***0.56***	N/A	N/A	N/A
	Correlation (ρ) with Respect to SWARM	***0.9817***	***0.97861***	***0.9251***	***0.9648***	***0.7581***	***0.9143***
CC$_{MST}$	F-Measure	0.41	0.42	0.52	N/A	N/A	N/A
	Correlation (ρ) with respect to UCLUST	0.7461	0.8143	0.8272	0.7417	0.6935	0.7974
	Correlation (ρ) with respect to SUMACLUST	0.7182	0.6903	0.8613	0.8293	0.6791	0.8213
	Correlation (ρ) with respect to SWARM	0.7914	0.7352	0.8904	0.7619	0.7372	0.8502

Table shows values of F-measures and Pearson Correlation Coefficient (ρ-value) of UCLUST, SUMACLUST, SWARM and their respective versions with Canopy clustering. F-measures is only available for synthetic datasets but not for real world datasets since no ground truths like known taxonomic profiles are available for them. ρ-value was calculated based on the taxonomy profiles at Genus level generated from clustered OTUs provided by a method and it's corresponding LSH based Canopy counterpart. Higher F-measures reflect better clustering by adhering to ground truth. Higher ρ-values reflect stronger correlation between taxonomic profiles.

Table III
RUNTIME COMPARISON (IN MINUTES)

Datasets			**Methods**											
Type	Title	# of Reads	UCLUST	Canopy	UCLUST with Canopy	Speed Up	SUMACLUST	Canopy	SUMACLUST with Canopy	Speed Up	SWARM	Canopy	SWARM with Canopy	Speed Up
Synthetic	Bokulich$_2$	6,938,836	12.71	4.81	***6.03***	***2.10x***	87.53	5.73	***33.03***	***2.65x***	128.12	5.29	***37.87***	***3.38x***
	Bokulich$_3$	3,594,237	8.91	2.91	***4.89***	***1.82x***	11.73	3.05	***7.89***	***1.48x***	9.10	2.98	***6.93***	***1.31x***
	Bokulich$_6$	250,903	1.21	0.49	***1.08***	***1.12x***	***1.28***	0.52	1.34	0.96	1.97	0.51	***1.29***	***1.53x***
Real World	Body Sites	886,630	2.12	1.03	***1.51***	***1.40x***	15.42	1.09	***7.88***	***1.95x***	3.64	0.98	***1.71***	***2.13x***
	Canadian Soil	2,966,053	9.55	2.18	***5.91***	***1.62x***	363.96	2.91	***81.18***	***4.48x***	97.50	3.11	***32.16***	***3.03x***
	Global Soil	9,252,764	72.47	21.38	***43.53***	***1.67x***	510.92	20.82	***66.27***	***7.70x***	269.07	21.17	***56.14***	***4.79x***

means lower $T1$ will yield better canopy assignment. From Figure 5a we can see that when $T1$'s range is in 0.4 to 0.6 our proposed approach provides better F-Scores. On the other hand, increasing $T2$ means comparatively *softer* tight-threshold for canopies. As a result instances will be prematurely assigned to canopies without waiting for best match. From Figure 5b we can see that when $T2$'s range is in 0.2 to 0.35 our proposed approach provides better F-Scores. Lower $T2$ may cause higher runtime since instances will continue to reappear until $T2$ is met.

VI. CONCLUSION AND FUTURE WORK

We propose a framework that can be used as pre-clustering for any accurate and relatively expensive clustering on large scale metagenomic datasets. Our proposed approach scales well with large datasets and provide significant reduction in computation time. We demonstrate that our approach provides similar outcome in terms of biodiversity metrics used in this study, scores based on ground truth and taxonomic correlation with corresponding relatively expensive cluster. Our approach takes advantage of the multi-core CPU systems by partitioning the large dataset roughly with fast and cheaper pairwise distance measure and then deploying comparatively expensive clustering in parallel which considers only data points that are inside the partition. We plan to develop standalone cluster mechanism for the canopies in future.

REFERENCES

[1] J. C. Venter, K. Remington, J. F. Heidelberg, A. L. Halpern, D. Rusch, J. A. Eisen, D. Wu, I. Paulsen, K. E. Nelson, W. Nelson *et al.*, "Environmental genome shotgun sequencing of the sargasso sea," *science*, vol. 304, no. 5667, pp. 66–74, 2004.

[2] Q. et al., "A human gut microbial gene catalogue established by metagenomic sequencing," *nature*, vol. 464, no. 7285, pp. 59–65, 2010.

[3] P. J. Turnbaugh, R. E. Ley, M. Hamady, C. Fraser-Liggett, R. Knight, and J. I. Gordon, "The human microbiome project: exploring the microbial part of ourselves in a changing world," *Nature*, vol. 449, no. 7164, p. 804, 2007.

[4] M. et al., "A whole-genome assembly of drosophila," *Science*, vol. 287, no. 5461, pp. 2196–2204, 2000.

[5] A. Charuvaka and H. Rangwala, "Evaluation of short read metagenomic assembly," *BMC genomics*, vol. 12, no. 2, p. 1, 2011.

[6] S. Chakravorty, D. Helb, M. Burday, N. Connell, and D. Alland, "A detailed analysis of 16s ribosomal rna gene segments for the diagnosis of pathogenic bacteria," *Journal of microbiological methods*, vol. 69, no. 2, pp. 330–339, 2007.

Table IV
BIODIVERSITY COMPARISON [FAITHS PHYLOGENETIC DIVERSITY METRIC (PD), SHANNON AND SIMPSON]

Methods	**Comparison Metric**	**Datasets**					
		Synthetic			*Real World*		
		Bokulich$_2$	Bokulich$_3$	Bokulich$_6$	Body Sites	Canadian Soil	Global Soil
UCLUST	PD Range	$[171.95 - 221.85]$	$[186.90 - 212.84]$	$[104.51 - 104.51]$	$[1.46 - 46.79]$	$[0.30 - 1352.73]$	$[2.98 - 3.29]$
	Shannon Range	$[2.52 - 3.51]$	$[2.43 - 3.54]$	$[5.87 - 5.87]$	$[0.29 - 7.67]$	$[2.32 - 10.85]$	$[1.84 - 8.30]$
	Simpson Range	$[0.55 - 0.75]$	$[0.55 - 0.76]$	$[0.96 - 0.96]$	$[0.049 - 0.98]$	$[0.80 - 0.99]$	$[0.0 - 0.98]$
CC$_{UCLUST}$	PD Range	$[164.79 - 217.72]$	$[169.41 - 198.36]$	$[109.33 - 109.33]$	$[2.37 - 47.13]$	$[0.52 - 1419.31]$	$[3.02 - 3.81]$
	Shannon Range	$[2.61 - 3.83]$	$[2.92 - 3.91]$	$[6.41 - 6.41]$	$[0.93 - 7.13]$	$[3.26 - 7.61]$	$[3.72 - 7.38]$
	Simpson Range	$[0.64 - 0.87]$	$[0.56 - 0.93]$	$[0.96 - 0.96]$	$[0.081 - 0.99]$	$[0.84 - 0.99]$	$[0.21 - 0.99]$
SUMACLUST	PD Range	$[106.00 - 162.78]$	$[142.85 - 174.19]$	$[89.22 - 89.22]$	$[0.93 - 39.47]$	$[0.59 - 1279.29]$	$[2.98 - 3.29]$
	Shannon Range	$[2.00 - 3.01]$	$[2.19 - 3.28]$	$[5.48 - 5.48]$	$[0.16 - 7.43]$	$[2.32 - 7.32]$	$[1.00 - 7.89]$
	Simpson Range	$[0.52 - 0.73]$	$[0.54 - 0.75]$	$[0.95 - 0.95]$	$[0.027 - 0.98]$	$[0.80 - 0.99]$	$[0.40 - 0.98]$
CC$_{SUMACLUST}$	PD Range	$[114.96 - 171.57]$	$[147.85 - 187.91]$	$[93.81 - 93.81]$	$[0.86 - 41.63]$	$[0.81 - 1292.34]$	$[1.37 - 4.89]$
	Shannon Range	$[2.51 - 3.94]$	$[2.96 - 4.11]$	$[5.94 - 5.94]$	$[1.21 - 7.13]$	$[3.12 - 7.79]$	$[2.17 - 7.25]$
	Simpson Range	$[0.68 - 0.79]$	$[0.51 - 0.74]$	$[0.96 - 0.96]$	$[0.06 - 0.99]$	$[0.88 - 0.99]$	$[0.23 - 0.98]$
SWARM	PD Range	$[18.37 - 24.73]$	$[17.36 - 19.81]$	$[30.84 - 30.84]$	$[1.44 - 28.66]$	$[0.54 - 706.57]$	$[5.79 - 6.18]$
	Shannon Range	$[2.98 - 3.91]$	$[2.01 - 3.04]$	$[5.03 - 5.03]$	$[0.28 - 7.63]$	$[1.0 - 7.79]$	$[1.66 - 7.81]$
	Simpson Range	$[0.70 - 0.82]$	$[0.53 - 0.74]$	$[0.95 - 0.95]$	$[0.05 - 0.98]$	$[0.50 - 0.99]$	$[0.00 - 0.99]$
CC$_{SWARM}$	PD Range	$[19.18 - 26.87]$	$[18.43 - 22.61]$	$[31.48 - 31.48]$	$[2.34 - 29.97]$	$[1.37 - 748.71]$	$[2.34 - 8.46]$
	Shannon Range	$[1.66 - 4.87]$	$[1.19 - 4.13]$	$[6.03 - 6.03]$	$[0.89 - 7.13]$	$[2.81 - 8.06]$	$[2.81 - 7.87]$
	Simpson Range	$[0.66 - 0.88]$	$[0.41 - 0.81]$	$[0.91 - 0.91]$	$[0.11 - 0.99]$	$[0.74 - 0.99]$	$[0.14 - 0.99]$
CC$_{MST}$	PD Range	$[10.43 - 89.61]$	$[12.83 - 92.41]$	$[63.21 - 63.21]$	$[1.46 - 34.15]$	$[2.73 - 824.62]$	$[1.81 - 5.37]$
	Shannon Range	$[0.96 - 3.91]$	$[1.21 - 3.16]$	$[4.91 - 4.91]$	$[2.02 - 6.51]$	$[2.72 - 7.83]$	$[2.17 - 6.29]$
	Simpson Range	$[0.14 - 0.68]$	$[0.32 - 0.73]$	$[0.92 - 0.92]$	$[0.07 - 0.94]$	$[0.58 - 0.94]$	$[0.14 - 0.98]$

Table shows ranges of values for Faiths Phylogeny Diversity (PD), Shannon and Simpson coefficient over all samples in a dataset in the format $[minimum - maximum]$. Most of these datasets contain multiple samples and Alpha diversity metrics like PD, Shannon and Simpson values are generated for each of these samples separately. Biodiversity metric values changes significantly over samples e.g. diversity from hair samples and teeth cavity are supposedly different. So instead of mean values this Table represents $[minimum - maximum]$ ranges of values a sample can take. Similar ranges reflect similar diversity.

[7] M. Blaxter, J. Mann, T. Chapman, F. Thomas, C. Whitton, R. Floyd, and E. Abebe, "Defining operational taxonomic units using dna barcode data," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 360, no. 1462, pp. 1935–1943, 2005.

[8] A. F. Koeppel and M. Wu, "Surprisingly extensive mixed phylogenetic and ecological signals among bacterial operational taxonomic units," *Nucleic acids research*, p. gkt241, 2013.

[9] T. S. Schmidt, J. F. M. Rodrigues, and C. Von Mering, "Ecological consistency of ssu rrna-based operational taxonomic units at a global scale," *PLoS Comput Biol*, vol. 10, no. 4, p. e1003594, 2014.

[10] Li, Weizhong and Godzik, Adam, "Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, 2006.

[11] Schloss, Patrick D. and Handelsman, Jo, "Introducing dotur, a computer program for defining operational taxonomic units and estimating species richness," *Applied and Environmental Microbiology*, vol. 71, no. 3, pp. 1501–1506, 2005.

[12] Schloss et al., "Introducing mothur: Open-source, platform-independent, community-supported software for describing and comparing microbial communities," *Applied and Environmental Microbiology*, vol. 75, no. 23, pp. 7537–7541, 2009.

[13] Sun, Yijun and Cai, Yunpeng and Liu, Li and Yu, Fahong and Farrell, Michael L. and McKendree, William and Farmerie, William, "Esprit: estimating species richness using large collections of 16s rrna pyrosequences," *Nucleic Acids Research*, 2009.

[14] Z. Rasheed, H. Rangwala, and D. Barbara, "Efficient clustering of metagenomic sequences using locality sensitive hashing." in *SDM*. SIAM, 2012, pp. 1023–1034.

[15] Z. Rasheed and H. Rangwala, *MC-MinH: Metagenome Clustering using Minwise based Hashing*, ch. 74, pp. 677–685. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611972832.75

[16] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher, "Min-wise independent permutations," *Journal of Computer and System Sciences*, vol. 60, no. 3, pp. 630–659, 2000.

[17] O. et al., "Mash: fast genome and metagenome distance estimation using minhash," *bioRxiv*, p. 029827, 2016.

[18] O. Tanaseichuk, J. Borneman, and T. Jiang, "Separating metagenomic short reads into genomes via clustering," *Algorithms for Molecular Biology*, vol. 7, no. 1, p. 1, 2012.

[19] Y.-W. Wu and Y. Ye, "A novel abundance-based algorithm for binning metagenomic sequences using l-tuples," *Journal of Computational Biology*, vol. 18, no. 3, pp. 523–534, 2011.

[20] S. Chatterji, I. Yamazaki, Z. Bai, and J. A. Eisen, "Compostbin: A dna composition-based algorithm for binning environmental shotgun reads," in *Annual International Conference on Research in Computational Molecular Biology*. Springer, 2008, pp. 17–28.

[21] M. Van Den Heuvel, R. Mandl, and H. H. Pol, “Normalized cut group clustering of resting-state fmri data,” *PloS one*, vol. 3, no. 4, p. e2001, 2008.

[22] Edgar, Robert C., “Search and clustering orders of magnitude faster than blast,” *Bioinformatics*, vol. 26, no. 19, pp. 2460–2461, 2010.

[23] F, Mah and T, Rognes and C, Quince and C, de Vargas and M, Dunthorn, “Swarm: robust and fast clustering method for amplicon-based studies,” *PeerJ*, vol. 2:e593, 2014.

[24] F. Mahé, T. Rognes, C. Quince, C. De Vargas, and M. Dunthorn, “Swarm v2: highly-scalable and high-resolution amplicon clustering,” *PeerJ, volume=3, pages=e1420, year=2015, publisher=PeerJ Inc.*

[25] C, Mercier and F, Boyer and A, Bonin and E, Coissac, “Sumatra and sumaclust: fast and exact comparison and clustering of sequences [submitted for publication],” *Available at https://git.metabarcoding.org/obitools/sumatra/wikis/home*, 2014.

[26] K. et al., “Open-source sequence clustering methods improve the state of the art,” *mSystems*, vol. 1, no. 1, pp. e00 003–15, 2016.

[27] McCallum, Andrew and Nigam, Kamal and Ungar, Lyle H., “Efficient clustering of high-dimensional data sets with application to reference matching,” in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’00. New York, NY, USA: ACM, 2000, pp. 169–178. [Online]. Available: http://doi.acm.org/10.1145/347090.347123

[28] Rajaraman, A. and Ullman, J, *Mining of Massive Datasets, Ch. 3*, 2010.

[29] Gionis, Aristides and Indyk, Piotr and Motwani, Rajeev, “Similarity search in high dimensions via hashing,” in *Proceedings of the 25th International Conference on Very Large Data Bases*, ser. VLDB ’99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 518–529. [Online]. Available: http://dl.acm.org/citation.cfm?id=645925.671516

[30] Indyk, Piotr and Motwani, Rajeev, “Approximate nearest neighbors: Towards removing the curse of dimensionality,” in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, ser. STOC ’98. New York, NY, USA: ACM, 1998, pp. 604–613. [Online]. Available: http://doi.acm.org/10.1145/276698.276876

[31] P. K. Jana and A. Naik, “An efficient minimum spanning tree based clustering algorithm,” pp. 1–5, 2009.

[32] M. Laszlo and S. Mukherjee, “Minimum spanning tree partitioning algorithm for microaggregation,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 7, pp. 902–911, 2005.

[33] Y. Xu, V. Olman, and D. Xu, “Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees,” *Bioinformatics*, vol. 18, no. 4, pp. 536–545, 2002.

[34] Charikar, Moses S., “Similarity estimation techniques from rounding algorithms,” in *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, ser. STOC ’02. New York, NY, USA: ACM, 2002, pp. 380–388. [Online]. Available: http://doi.acm.org/10.1145/509907.509965

[35] Bokulich, NA and Subramanian, S and Faith JJ and Gevers, D and Gordon, JI and Knight, R and Mills, DA and Caporaso, JG, “Quality-filtering vastly improves diversity estimates from illumina amplicon sequencing,” *Nat Methods*, vol. 10, pp. 57–59, 2013.

[36] JD, Neufeld and K, Engel and J, Cheng and G, Moreno-Hagelsieb and DR, Rose and TC,Charles, “Open resource metagenomics: a model for sharing metagenomic libraries,” *Standards in Genomic Sciences*, vol. 5, pp. 203–210, 2011.

[37] Costello, Elizabeth K. and Lauber, Christian L. and Hamady, Micah and Fierer, Noah and Gordon, Jeffrey I. and Knight, Rob, “Bacterial community variation in human body habitats across space and time,” *Science*, vol. 326, no. 5960, pp. 1694–1697, 2009.

[38] Ramirez et al., “Biogeographic patterns in below-ground diversity in new york city’s central park are similar to those observed globally,” *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 281, no. 1795, 2014.

[39] Kopylova et al., “Open-source sequence clustering methods improve the state of the art,” *mSystems*, vol. 1, no. 1, 2016.

[40] Faith, Daniel P, “Conservation evaluation and phylogenetic diversity,” *Biological conservation*, vol. 61, no. 1, pp. 1–10, 1992.

[41] Caporaso et al., “Qiime allows analysis of high-throughput community sequencing data,” *Nat Meth*, vol. 7, no. 5, pp. 335–336, 2010.

[42] Caporaso, J. Gregory and Bittinger, Kyle and Bushman, Frederic D. and DeSantis, Todd Z. and Andersen, Gary L. and Knight, Rob, “Pynast: a flexible tool for aligning sequences to a template alignment,” *Bioinformatics*, vol. 26, no. 2, pp. 266–267, 2010.