Ananke 2: Memory-efficient, progressive clustering of large microbial data sets

1st Michael Hall
Faculty of Graduate Studies
Dalhousie University
Halifax, Nova Scotia
mike.hall@dal.ca

2nd Robert Beiko
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia
rbeiko@cs.dal.ca

***Abstract*—As data sets grow and combine and the capacity to collect data increases, resource efficient algorithms are becoming more necessary for handling microbial ecology data. Clustering, exploring, and visualizing data becomes difficult without the aid of significant computing resources. In this paper, we present Ananke 2, an algorithm for dynamically and progressively clustering large sets of microbial features. Our algorithm uses bloom filters to store the relationships between microbial features (taxonomic unit counts, amplicon counts, functional gene counts, etc.). These succinct, probabilistic data structures allow the network structure of the microbial features to be stored in a memory-efficient way, enabling the clustering and exploration of very large feature sets. Ananke 2 is particularly suited to clustering time-series data, and supports time-series specific distance measures such as short time-series distance and dynamic time warping.**

***Index Terms*—bioinformatics, microbiology, amplicon sequencing, clustering, bloom filters**

I. Introduction

Researchers in the microbiome fields are capable of collecting quantitative data on biological features, such as taxonomic units or groups or functional genes, at steadily increasing rates. As a result of an "explosion in sequencing data", there has been a proliferation of tools to process, analyze, and manage microbiome data [Narayan et al., 2020]. As time-series data sets continue to grow, meta-analyses of combined data sets are created, and overall sequencing capabilities increase, the need for efficient algorithms and data management strategies increases.

One common concept in microbiome data science is data clustering, which can be done to reduce data sets to more computationally manageable sizes or to identify and/or collapse groups of similar objects. Sequence clustering, such as the creation of operational taxonomic units (OTUs) or amplicon sequence variance (ASVs), is done to collapse large sequence data sets into a set of similar sequences that is manageable for downstream analysis such as phylogenetic tree building and ordinations. Taxa, functional genes, or samples can all be clustered to identify and investigate similar groups of these objects. Techniques range from hierarchical clustering, to graph-based, and machine learning methods.

In a previous paper, we presented Ananke [Hall et al., 2017], a computational tool for clustering amplicon sequences and taxonomic units that were collected as a part of a time series. In that analysis, we showed that a time-adjusted distance measure, the short time-series distance, performed well for aggregating time series by overall dynamics and producing meaningful time-series clusters. We also demonstrated that the temporal dynamics of the sequences that comprised an OTU were not always consistent, suggesting that sequence similarity algorithms were grouping different ecotypes into a single taxonomic unit. This has the impact of potentially obscuring or altering true temporal dynamics, and has been noted by others such as Tikhonov et al. [2015].

The presented work builds on the concepts and goals of the first Ananke project, but aims to be more memory-efficient in order to support the feature sets of larger projects and meta-analyses. The second iteration of this algorithm also aims to be more robust, supporting partial clustering and interrupting large computations to explore processed data.

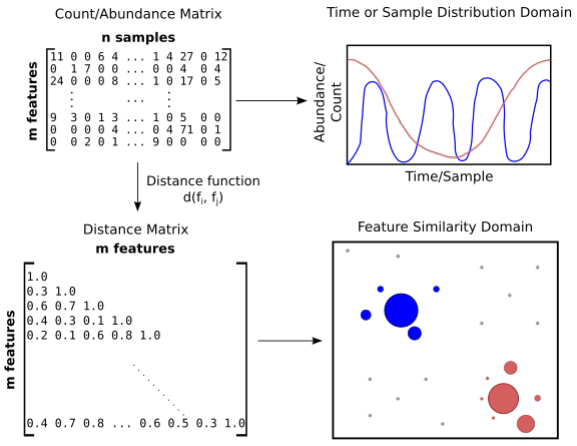

Fig. 1. Input data consists of feature counts or abundances across a series of samples that possibly form a time-series. A distance measure is used to compute the distances between all pairs of features. This distance matrix can be used to explore the sequences in a similarity domain, such as by clustering.

When clustering biological features, the general process is to take the numeric count or abundance matrix (normalized as appropriate) and compute a distance matrix with a given distance function (Figure 1). If the number of features is low (in the low

thousands), then modern hardware can compute and store the entire m features by m features distance matrix. The $O(m^2$ storage complexity of the feature distance matrix means that each additional feature adds an exponentially-growing amount of memory required. Unlike the count/abundance matrix which is often highly sparse, a distance matrix tends to be very dense, meaning a more succinct representation will require more creativity to generate. We harness bloom filters, a succinct probabilistic data structure, in order to generate a useful representation of the feature distance matrix in a smaller, fixed-memory space. Alongside the DBSCAN clustering algorithm Ester et al. [1996] used in the original Ananke algorithm, the improved efficiencies allow more expensive distance measures, such as dynamic time warping (DTW) Berndt and Clifford [1994] to be used to generate time-series clusters, while integration with the `scipy` software package provides access to numerous ecologically-relevant distance measures for time-agnostic applications.

II. Methods

Ananke 2 is open-source software available online at GitHub (https://github.com/mwhall/q2_ananke).

As mentioned previously, the distance matrix computed between microbial features can quickly become too large for memory, especially as the number of features is generally much larger than the number of samples. Our algorithm employs succinct, probabilistic data structures to capture and query the neighbour relationships of the features in a small memory footprint. The probabilistic data structure used by Ananke 2, the bloom filter, are objects that store the membership of items in a set.

The Ananke 2 algorithm involves computing the distances between microbial features and storing pairs of feature identifiers in bloom filters when they are neighbours at a given threshold. This captures the network structures of the features in a succinct bit-string, which can be easily and efficiently loaded from disk to memory. It is a light-weight, practical representation of the relationships between microbial features, and enables useful data exploration and clustering strategies for very large feature sets.

The algorithm initializes with a `m` features $\times$ `n` sample count or abundance table (Figure 1). Features are considered neighbours if the computed distance between them is below a specified threshold.

Features are added progressively to the data structure, allowing it to expand its scope as time and computational resources allow. Ananke 2 is designed to be interrupted to allow immediate exploration of a partially-computed set of features. Features can be added to the computation queue by several criteria: A) Abundance, B) Taxonomy, C) Search, and D) Cluster expansion (Figure 2). In many situations, the most abundant features are the most likely to be of importance, so a good strategy is to begin by computing the relationships between the top N abundant features or by using some other abundance cut-off. This captures the neighbourhoods of abundant features and can be used to explore the partially-computed

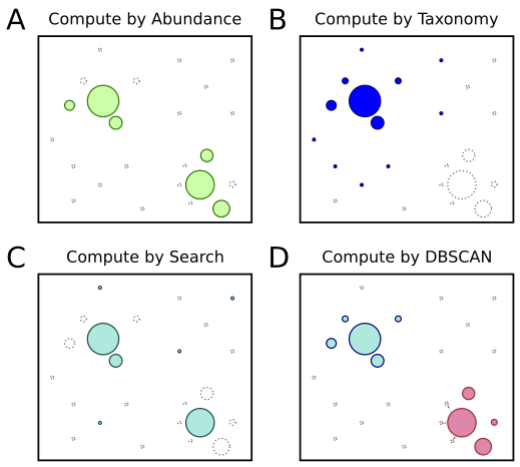

Fig. 2. Circles represent features arranged in a similarity domain. Filled circles are features that have had their distances computed, while dotted circles represent features that are in the full data set, but are not yet computed and will not be part of any clusters until they are. A larger diameter implies a larger abundance. A) The most abundant features (green) are computed, and low abundance features remain uncomputed. B) If the features are sequences that can be taxonomically classified, all features of a given taxonomic group (e.g., blue) can be computed. C) Features can be queried by name and, if not already computed, are computed at query time and do not have to be recomputed subsequently. D) The features belonging to a specified DBSCAN cluster can be computed against all outstanding, uncomputed features to identify undetected neighbours in the full data set and compute all distances. Blue and pink represent DBSCAN clusters of features, and pink has been expanded and three low abundance features were detected as neighbours and added to the cluster.

similarity space that includes those features. If the features are amplicon sequences, taxonomic metadata can be provided to ensure all features of a given taxonomy are computed. If a feature is queried by name, then it will be added to the computation queue if not already added. Finally, a cluster of pre-computed features can be specified and used to search the uncomputed set of features for any similarly distributed sequences, and expanded by their inclusion.

Once distances are computed, the data can be clustered to identify similar groups of features. The DBSCAN algorithm can be computed quickly when the `neighbours(`f_i`,`f_j`,`ϵ`)` operation can be returned in constant time. The bloom filter structures store the neighbour pairs, and set membership from a bloom filter is returned in constant $O(1)$ time. The Ananke 2 algorithm requires a discrete range of distance cut-offs, the DBSCAN ϵ parameter, and uses bloom filters to store the pairs of features that are neighbours at each ϵ value in that range (Figure 3). Bloom filters have a capacity after which too many items have been added such that the false positive rate is higher than a specified tolerance due to the presence of too many flipped one bits in the bit string. When a bloom filter hits this capacity for a given ϵ, it is removed and deleted. Often the bloom filters fill because the specified ϵ is large and produces nearly fully-connected graphs of features, but the user can increase a memory multiplier parameter to recompute with more capacity.

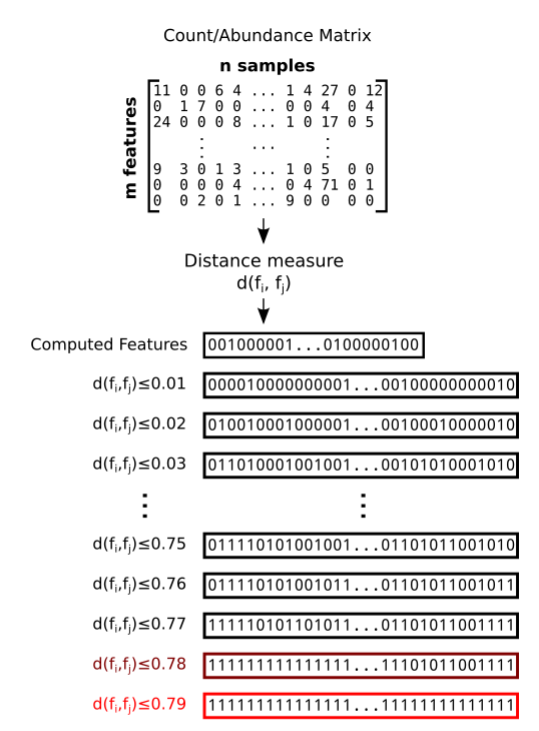

Fig. 3. Features are added to the data structure in a user-specified priority. Features that have been computed are added to a smaller bloom filter, and the pair f_i, f_j are added to all ϵ value bloom filters when the distance between the features is less than or equal to that filter's threshold. When a bloom filter is full (red), its false positive rate will be too high and it will be removed.

III. Results

The clusters can be explored by plotting as time-series. If a feature name is provided, Ananke 2 will return the clusters over a range of ϵ distance values. As that ϵ becomes larger, the clusters that are being returned are larger. Eventually, as ϵ is increased the clusters will merge and become harder to interpret. But for smaller ϵ values, the clusters can show clear dynamics. Figure 4 shows an example time-series cluster from the Bedford Basin consisting of 10 distinct variants.

IV. Conclusions

We present the second version of Ananke, an algorithm for clustering microbiome data. This version uses probabilistic data structures to map large interaction networks onto small in-memory objects. These data structures can be queried for fast clustering of very large similarity spaces. Ananke 2 is useful for clustering time-series data, or any features by more general distance measures. The progressive clustering strategy allows very large data sets to be worked with as they are computed, revealing areas of a similarity domain as they become relevant

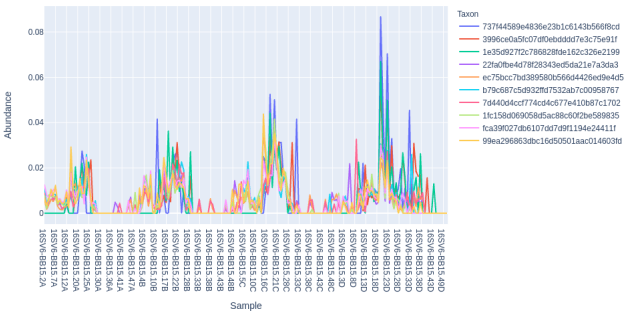

Fig. 4. Example of a time-series cluster from a Bedford Basin data set. Similarity was measured by the cosine distance.

to the user, and making it faster and easier to begin working with larger data sets.

References

Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

Michael W Hall, Robin R Rohwer, Jonathan Perrie, Katherine D McMahon, and Robert G Beiko. Ananke: temporal clustering reveals ecological dynamics of microbial communities. *PeerJ*, 5:e3812, 2017.

Aditya Narayan, Ajeet Singh, and Shailesh Kumar. Understanding microbiome science through big data analysis. In *Metagenomic Systems Biology*, pages 55–74. Springer, 2020.

Mikhail Tikhonov, Robert W Leach, and Ned S Wingreen. Interpreting 16s metagenomic data without clustering to achieve sub-otu resolution. *The ISME journal*, 9(1):68–80, 2015.