TECHNICAL NOTE

Libra: scalable *k*-mer–based tool for massive all-vs-all metagenome comparisons

Illyoung Choi [1], Alise J. Ponsero [2], Matthew Bomhoff [2],
Ken Youens-Clark [2], John H. Hartman [1,*] and Bonnie L. Hurwitz [2,3,*]

[1]Department of Computer Science, University of Arizona, 1040 E. 4th Street, Tucson, Arizona, 85721, USA,
[2]Department of Biosystems Engineering, University of Arizona, 1177 E. 4th Street, Tucson, Arizona, 85721, USA
and [3]BIO5 Institute, University of Arizona, 1657 E. Helen Street, Tucson, Arizona, 85719, USA

*Correspondence address. 1177 E. 4th Street, Tucson, Arizona, 85721, USA. Bonnie L. Hurwitz, E-mail:
bhurwitz@email.arizona.edu http://orcid.org/0000-0001-8699-957X; John H. Hartman, E-mail:
jhh@cs.arizona.edu http://orcid.org/0000-0002-5557-0604

Abstract

Background: Shotgun metagenomics provides powerful insights into microbial community biodiversity and function. Yet, inferences from metagenomic studies are often limited by dataset size and complexity and are restricted by the availability and completeness of existing databases. *De novo* comparative metagenomics enables the comparison of metagenomes based on their total genetic content. **Results:** We developed a tool called Libra that performs an all-vs-all comparison of metagenomes for precise clustering based on their *k*-mer content. Libra uses a scalable Hadoop framework for massive metagenome comparisons, Cosine Similarity for calculating the distance using sequence composition and abundance while normalizing for sequencing depth, and a web-based implementation in iMicrobe (http://imicrobe.us) that uses the CyVerse advanced cyberinfrastructure to promote broad use of the tool by the scientific community. **Conclusions:** A comparison of Libra to equivalent tools using both simulated and real metagenomic datasets, ranging from 80 million to 4.2 billion reads, reveals that methods commonly implemented to reduce compute time for large datasets, such as data reduction, read count normalization, and presence/absence distance metrics, greatly diminish the resolution of large-scale comparative analyses. In contrast, Libra uses all of the reads to calculate *k*-mer abundance in a Hadoop architecture that can scale to any size dataset to enable global-scale analyses and link microbial signatures to biological processes.

Keywords: metagenomics; Hadoop; *k*-mer; distance metrics; clustering

Introduction

Over the last decade, scientists have generated petabytes of genomic data to uncover the role of microbes in dynamic living systems. Yet, to understand the underlying biological principles that guide the distribution of microbial communities, massive 'omics datasets need to be compared with environmental factors to find linkages across space and time. One of the greatest challenges in these endeavors has been in documenting and analyzing unexplored genetic diversity in wild microbial communities. For example, fewer than 60% of 40 million non-redundant genes from the Global Ocean Survey and the Tara Oceans Expeditions match known proteins in bacteria [1, 2]. Other microorganisms such as viruses or pico-eukaryotes that are important to ocean ecosystems are even less well defined (e.g., <7% of reads from viromes match known proteins [3]). This is largely due to the fact that these organisms are unculturable and reference genomes do not exist in public data repositories. Thus, genome sequences from metagenomic data await better taxonomic and functional definition. Consequently, even advanced tools such as *k*-mer–based classifiers that rapidly assign metage-

nomic reads to known microbes miss "microbial dark matter" that comprises a significant proportion of metagenomes [4–6].

De novo comparative metagenomics offers a path forward

In order to examine the complete genomic content, metagenomic samples can be compared using their sequence signature (or frequency of k-mers) (list of tools available in Supplementary Table S1A). This approach relies on three core tenets of k-mer–based analytics: (i) closely related organisms share k-mer profiles and cluster together, making taxonomic assignment unnecessary [7, 8]; (ii) k-mer frequency is correlated with the abundance of an organism [9]; and (iii) k-mers of sufficient length can be used to distinguish specific organisms [10]. In 2012, the Compareads [11] method was proposed, followed by Commet [12]. Both of these tools compute the number of shared reads between metagenomes using a k-mer–based read similarity measure. The number of shared reads between datasets is then used to compute a Jaccard distance between samples.

Given the computational intensity of all-vs-all sequence analysis, several other methods have been employed to reduce the dimensionality of metagenomes and speed up analyses by creating unique k-mer sets and computing the genetic distance between pairs of metagenomes, such as MetaFast [13] and Mash [14]. The fastest of these methods, Mash [15], indexes samples by unique k-mers to create size-reduced sketches and compares these sketches using the MinHash algorithm [16] for computing a genetic distance using Jaccard similarity. Yet, the tradeoff for speed is that samples are reduced to a subset of unique k-mers (1 k by default) that may lead to an unrepresentative k-mer profile of the samples. Further, given that Mash uses Jaccard similarity, only the genetic distance between samples is accounted for (or genetic content in microbial communities) without considering abundance (dominant vs rare organisms in the sample), which is central to microbial ecology and ecosystem processes [17]. Sourmash [18], a toolkit for manipulating MinHash sketches, uses the same underlying algorithm and distance metric as Mash and therefore has the same limitations.

Recently, Simka [15] was developed to compute a distance matrix between metagenomes by dividing the input datasets into abundance vectors from subsets of k-mers, then rejoining the resulting abundances in a cumulative distance matrix. The methodology can be parallelized to execute the analyses on a high-performance computing cluster (HPC). Simka also provides various ecological distance metrics to let the user choose the metric most relevant to their analysis. However, the computational time varies based on the distance metric, where some distances scale linearly and other distances metrics, such as Jensen-Shannon, scale quadratically as additional samples are added [15]. Moreover, Simka normalizes datasets in an all-vs-all comparison by reducing the depth of sequencing for all samples to the least common denominator, therefore, decreasing the resolution of the datasets. Lastly, computing k-mer analytics using HPC is subject to reduced fault tolerance for massive datasets. A framework to compare one metagenome to a set of metagenomes on a high-performance computing system called DSM [19] has also been proposed; however, this tool is limited to retrieval tasks and does not provide an all-vs-all sequence analysis.

Scaling sequence analysis using big data analytics via Hadoop

Hadoop is an attractive platform for performing large-scale sequence analysis because it provides a distributed file system and distributed computation for analyzing massive amounts of data. Hadoop clusters are comprised of commodity servers so that the processing power increases as more computing resources are added. Hadoop also offers a high-level programming abstraction, called MapReduce [20], that greatly simplifies the implementation of new analytical tools and a high-performance distributed file system (HDFS) for storing datasets. Programmers do not need specialized training in distributed systems and networking to implement distributed programs using MapReduce. Hadoop also provides fault-tolerance by default. When a Hadoop node fails, Hadoop reassigns the failed node's tasks to another node containing a redundant copy of the data those tasks were processing. This differs from HPC where schedulers track failed nodes and either restart the failed computation from the most recent checkpoint or from the beginning if checkpointing was not used. Thus, using a Hadoop infrastructure ensures that computations and data are protected even in the event of hardware failures. These benefits have led to new analytic tools based on Hadoop, making Hadoop a *de facto* standard in large-scale data analysis. In metagenomics, the development of efficient and inexpensive high-throughput sequencing technologies has led to a rapid increase in the amount of sequence data for studying microbes in diverse environments. However, to date, only Hadoop-enabled genomic or k-mer counting tools exist, and no comparative metagenomics tools are available (Supplementary Table S1B).

Existing big data algorithms compare reads to limited genomic reference data

Recent progress has been made in translating bioinformatics algorithms to big data architectures to overcome scalability issues. Thus far, these algorithms compare large-scale next-generation sequence (NGS) datasets to reference genomic datasets and replace computationally intensive algorithms such as sequence alignment [21], genetic variant detection [22, 23], ortholog detection [24], differential gene expression [25, 26], or short-read mapping [27–30] (Supplementary Table S1B). For example, BlastReduce and CloudBurst are parallel sequence mapping tools based on Hadoop MapReduce [28, 29]. These tools, however, implement a query-to-a-reference approach that is inefficient for all-vs-all analyses of reads from metagenomes. Other algorithms such as BioPig [31] and Bloomfish [32] generate an index of sequence data for later partial sequence search and k-mer counting using Hadoop [33] (Supplementary Table S1B). Also, some of these tools adopt traditional sequence indexing techniques such as a suffix array that is inefficient in reading and indexing data in HDFS, thus reducing performance. Moreover, neither tool offers an end-to-end solution for comparing metagenomes consisting of data distribution on a Hadoop cluster, k-mer indexing and counting, distance matrix computation, and visualization. Finally, none of these tools are enabled in an advanced cyberinfrastructure where users can compute analyses in a simple web-based platform (Supplementary Table S1B).

Libra: a tool for scalable all-vs-all sequence analysis in an advanced cyberinfrastructure

Here, we describe a scalable algorithm called Libra that is capable of performing all-vs-all sequence analysis using Hadoop MapReduce (SciCrunch.org tool reference ID SCR.016608). We demonstrate for the first time that Hadoop MapReduce can be applied to all-vs-all sequence comparisons of large-scale metagenomic datasets comprised of mixed microbial communities. We demonstrate that Cosine Similarity, which is widely used in document clustering and information retrieval, is a good distance metric for comparing datasets to consider genetic distance and microbial abundance simultaneously, along with widely accepted distance metrics in biology such as Bray-Curtis [34] and Jensen-Shannon [35]. We validate this distance metric using simulated metagenomes (from both short- and long-read technologies) to show that Libra has exceptional sensitivity in distinguishing complex mixed microbiomes. Next, we show Libra's ability to distinguish metagenomes by both community composition and abundance using 48 samples (16S rRNA and whole-genome shotgun sequencing [WGS]) from the Human Microbiome Project (HMP) and the simulated Critical Assessment of Metagenome Interpretation (CAMI) "toy" Pacific Biosciences (PacBio) dataset across diverse body sites and compare the results to Mash and Simka. Finally, we show that Libra can scale to massive global-scale datasets by examining viral diversity in 43 Tara Ocean Viromes (TOVs) from the 2009–2011 Expedition [36] that represent 26 sites containing about 4.2 billion reads. We show for the first time that viral communities in the ocean are similar across temperature gradients, irrespective of their location in the ocean. The resulting data demonstrate that Libra provides accurate, efficient, and scalable computation for comparative metagenomics that can be used to discern global patterns in microbial ecology.

To promote the broad use of the Libra algorithm, we developed a web-based tool in iMicrobe [37] where users can run Libra using data in their free CyVerse [38, 39] account or use datasets that are integrated into the iMicrobe Data Commons. These analyses are fundamental for determining relationships among diverse metagenomes to inform follow-up analyses on microbial-driven biological processes.

Data Description

Staggered mock community

We performed metagenomic shotgun sequencing on a staggered mock community obtained from the Human Microbiome Consortium (HM-277D). The staggered mock community is comprised of genomic DNA from genera commonly found on or within the human body, consisting of 1,000 to 1,000,000,000 16S rRNA gene copies per organism per aliquot. The resulting DNA was subjected to whole-genome sequencing as follows. Mixtures were diluted to a final concentration of 1 ng/μL and used to generate whole genome sequencing libraries with the Ion Xpress Plug Fragment Library Kit and manual #MAN0009847, revC (Thermo Fisher Scientific, Waltham, MA). Briefly, 10 ng of bacterial DNA was sheared using the Ion Shear enzymatic reaction for 12 minutes and Ion Xpress bar code adapters ligated following end repair. Following bar code ligation, libraries were amplified using the manufacturer's supplied Library Amplification primers and recommended conditions. Amplified libraries were size-selected to ~200 base pairs using the Invitrogen E-gel Size Select Agarose cassettes as outlined in the Ion Xpress

manual and quantitated with the Ion Universal Library quantitation kit. Equimolar amounts of the library were added to an Ion PI Template OT2 200 kit V3. The resulting templated beads were enriched with the Ion OneTouch ES system and quantitated with the Qubit Ion Sphere Quality Control kit (Life Technologies) on a Qubit 3.0 fluorometer (Qubit, New York). Enriched templated beads were loaded onto an Ion PI V2 chip and sequenced according to the manufacturer's protocol using the Ion PI Sequencing 200 kit V3 on an Ion Torrent Proton sequencer. The sequence data, comprised of ~80 million reads, have been deposited into the National Center for Biotechnology Information Sequence Read Archive under accession SRP115095 under project accession PRJNA397434.

Simulated data derived from the staggered mock community

The resulting sequence data from the staggered mock community (~80 million reads) were used to develop simulated metagenomes to test the effects of varying read depth and of the composition and abundance of organisms in mixed metagenomes [40]. To examine read depth (in terms of raw read counts and file size), we used the known staggered mock community abundance profile to generate a simulated metagenome using GemSim [41] of 2 million reads (454 sequencing) and duplicated the dataset 2x, 5x, and 10x. We also simulated the effects of sequencing a metagenome more deeply using GemSim [41] to generate simulated metagenomes with 0.5, 1, 5, and 10 million reads based on the relative abundance of organisms in the staggered mock community. Next, we developed four simulated metagenomes to test the effect of changing the dominant organism abundance and genetic composition, including 10 million reads from the staggered mock community (mock 1), the mock community with alterations in a few abundant species (mock 2), the mock community with many alterations in abundant species (mock 3), and mock 3 with additional sequences from archaea to further alter the genetic composition (mock 4) as described in Supplementary Table S2. The same community profiles were used to generate paired-end Illumina dataset (100 million reads), using GemSim (Illumina v4 error model). Finally, using SimLord [42], the community profiles were used to generate simulated third-generation sequencing datasets (PacBio single-molecule real-time sequencing, 1 million reads). SimLord default parameters were used to generate those simulated datasets. All simulated datasets are available in iMicrobe [37] under project 265 and under DOI [40].

Human microbiome 16S rRNA gene amplicons and WGS reads

Human microbiome datasets were downloaded from the National Institutes of Health Human Microbiome Project [43] including 48 samples from 5 body sites including urogenital (posterior fornix), gastrointestinal (stool), oral (buccal mucosa, supragingival plaque, tongue dorsum), airways (anterior nares), and skin (retroauricular crease left and right) (See Supplementary Table S3). Matched datasets consisting of 16S rRNA reads WGS reads, and WGS assembled contigs were downloaded from the 16S trimmed dataset and the HMIWGS/HMASM dataset, respectively. For the WGS reads dataset, the analysis was run on the paired 1 read file.

Tara ocean viromes

TOVs were downloaded from European Nucleotide Archive at the European Molecular Biology Laboratory (EMBL) and consisted of 43 viromes from 43 samples at 26 locations across the world's oceans collected during the Tara Oceans (2009–2012) scientific expedition (Supplementary Table S4) [36]. Metadata for the samples were downloaded from PANGAEA [44]. These samples were derived from multiple depths including 16 surface samples (5–6 meters), 18 deep chlorophyll maximum (DCM) samples (17–148 meters), and 1 mesopelagic sample (791 meters). Quality-control procedures were applied according to the methods described by Brum and colleagues [36].

CAMI human microbiome project toy dataset

The HMP toy dataset from the CAMI 2nd Challenge was downloaded from their website [45]. This dataset is composed of 49 simulated PacBio reads from five different body sites of the human host, namely, gastrointestinal tract, oral cavity, airways, skin, and urogenital tract.

Results and Discussion

Libra computational strategy

Libra uses Hadoop MapReduce to perform massive all-vs-all sequence comparisons between next-generation sequence datasets. Libra uses a scalable algorithm and efficient resource usage to make all-vs-all comparisons feasible on large datasets. Hadoop allows parallel computation over distributed computing resources via its simple programming interface called *MapReduce*, while hiding much of the complexity of distributed computing (e.g., node failures) for robust fault-tolerant computation. Taking advantage of Hadoop, Libra can scale to larger input datasets and more computing resources. Furthermore, many cloud providers such as Amazon and Google offer Hadoop clusters on a pay-as-you-go basis, allowing scientists to scale their Libra computations to match their datasets and budgets.

Libra is implemented using three different MapReduce jobs: (1) *k*-mer histogram construction, (2) inverted index construction, and (3) distance matrix computation. Figure 1 shows a workflow of the Libra algorithm.

Libra distance computation

Jaccard and Bray-Curtis distance have been extensively used to compare metagenomes based on their sequence signature [13–15]. While Mash only computes the Jaccard distance between samples, Simka and Libra implement several classic ecology distances, allowing the user to choose the best-suited distance for the considered dataset [15]. Libra provides three distance metrics—Cosine Similarity, Bray-Curtis, and Jensen-Shannon. Here, we demonstrate Cosine Similarity as the default distance metric. This distance uses a vector space model to compute the distance between two NGS samples based on their *k*-mer composition and abundance, while simultaneously normalizing for sequencing depth. Cosine Similarity is widely used in document clustering and information retrieval. This distance metric was previously used to evaluate the accuracy of methods to reconstruct genomes from "virtual metagenomes" derived from 16S rRNA data based on shared Kyoto Encyclopedia of Genes and Genomes orthologous gene counts [46] but has not been applied in analyzing sequence signatures between metagenomes. Libra users can also weight *k*-mers based on their abundance (using Boolean weighting, natural weighting, and logarithmic weighting) to account for differences in microbial community composition and sequencing effort as detailed below.

Cosine Similarity allows for an accurate and normalized comparison of metagenomes

We explored the effects of varying (1) the size of the datasets, (2) depth of sequencing, (3) the abundance of dominant microbes in the community, and (4) genetic composition of the community by adding in an entirely new organism (in our case, we added archaea). We constructed simulated metagenomes and compared Libra's distance based on the Cosine Similarity against those from Mash and Simka. Simulated datasets were derived from genomic DNA from a staggered mock community of bacteria obtained from the Human Microbiome Consortium and sequenced deeply using the Ion Torrent sequencing platform (80 million reads; see the Methods section).

First, we examined the effect of the size of the dataset by using GemSim [41] to obtain a simulated metagenome composed of 1 million reads (454 sequencing) from the mock community and duplicated that dataset 2x and 10x. Overall, we found that altering the size of the metagenome (by duplicating the data) had no effect on the distance between metagenomes for Mash, Simka, or Libra. In each case, the distance of the duplicated datasets to the 1x mock community was less than 0.0001 (data not shown).

Because metagenomes do not scale exactly with size and instead have an increasing representation of low-abundance organisms, we created a second simulated dataset from the mock community using GemSim [41] 0.5, 1, 5, and 10 million reads (454 sequencing) to mimic the effect of reducing the sequencing. Given the abundance of organisms in the mock community, the 0.5 M read dataset is mainly comprised of dominant species. Because Simka normalizes all samples to the lowest read count, no changes between samples were measurable when using Jaccard and Bray-Curtis distances (Fig. 2A). In contrast, Mash and Libra (natural weighting) take into account all of the reads in the metagenomes; therefore, they measure a larger difference when you compare the smallest (0.5 M read sample) and largest (10 million read sample). These results suggest that Libra (natural weighting) and Mash are appropriate for comparing datasets at different sequencing depths, whereas using Simka could lead to undesired effects.

In addition to natural variation in population-level abundances, artifacts from sequencing can result in high-abundance *k*-mers. Libra allows users to select the optimal methodology for weighting high-abundance *k*-mers in their datasets including Boolean, natural, and logarithmic. These options for weighting *k*-mers are important for different biological scenarios as described below and shown in simulated datasets. To examine the effect of weighting, we compared and contrasted the natural and logarithmic weight in Libra with other distances obtained from Mash and Simka (Jaccard and Bray-Curtis). We also examined the effect of adding an entirely new species by spiking a simulated dataset with sequences derived from archaea (that were not present in the mock community). The simulated datasets (454 technology) were comprised of the staggered mock community (mock 1), the mock community with alterations in a few abundant species (mock 2), the mock community with many alterations in abundant species (mock 3), and mock 3 with additional sequences from archaea to alter the genetic composition of the community (mock 4) (see Supplementary Table S2). The resulting data showed that Libra (logarithmic weighting) shows
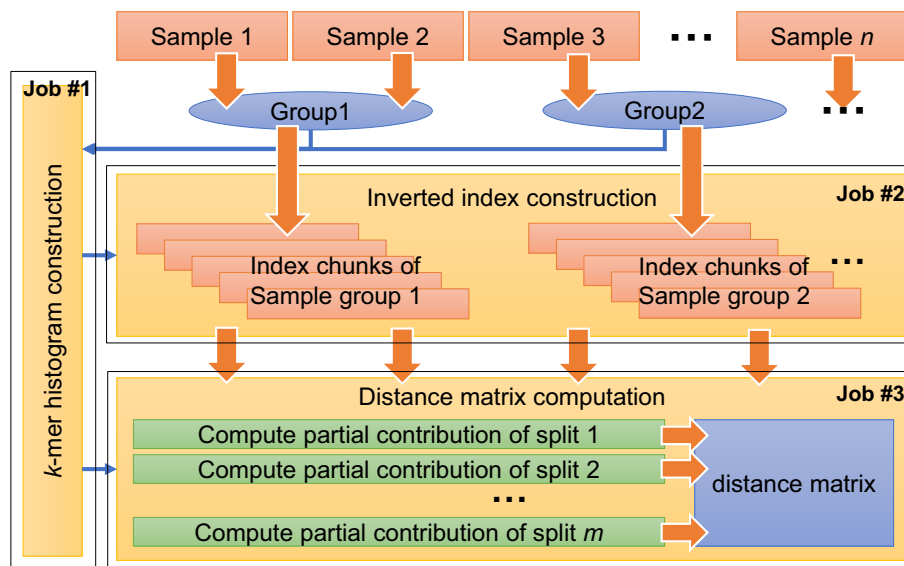
Figure 1: The Libra workflow. Libra consists of three MapReduce jobs (yellow boxes): (1) Libra constructs a *k*-mer histogram of the input samples for load-balancing. The *k*-mer histogram of the input samples is computed in parallel by running multiple Map tasks and a Reduce task that combines their results. (2) Libra constructs the inverted index in parallel. In the Map phase, a separate Map task is spawned for every data block in the input sample files. Each Map task generates *k*-mers from the sequences stored in a data block then passes them to the Reduce tasks. Each Reduce task then counts *k*-mers it receives and produces an index chunk. (3) In the distance matrix computation, the work is split by partitioning the *k*-mer space at the beginning of a MapReduce job. The *k*-mer histogram files for input samples are loaded, and the *k*-mer space is partitioned according to the *k*-mer distributions. A separate Map task is spawned for each partition to perform the computation in parallel and merged to produce the complete distance matrix.
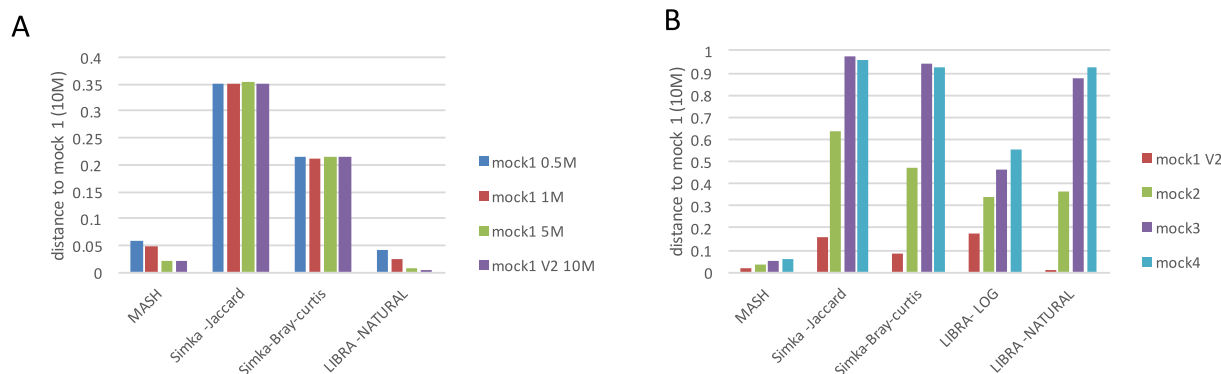

Figure 2: Analysis of simulated metagenomes using Mash, Simka, and Libra. **(A)** Distance to staggered mock community simulated metagenome composed of 10 million reads (mock 1 10 M), for simulated metagenomes of same community sequenced at various depth. Simulated metagenomes (454 sequencing) were obtained using GemSim and the known abundance profile of the staggered mock community (see Supplementary Table S2). In order to mimic various sequencing depths, the simulated metagenomes were generated at 0.5, 1, 5, or 10 million reads (noted mock 1 0.5 M; mock 1 1 M; mock 1 5 M; mock 1V2 10 M). The distances between the four simulated metagenomes and a 10 million read simulated metagenome (mock 1 10 M) was computed using Mash, Simka (Jaccard and Bray-Curtis distance), and Libra (natural weighting). **(B)** Distance to staggered mock community simulated metagenome (mock 1) for simulated metagenomes from increasingly distant communities. The mock 1 relies on the known abundance profile from the staggered mock community. The mock 2 community profile was obtained by randomly inverting three species abundance from mock 1 profile. The mock 3 profile was obtained by randomly inverting two species abundances from mock 2 profile. Finally, a mock 4 profile was obtained by adding high-abundance archeal genomes not present in any the other mock communities. Simulated metagenomes (454 sequencing) were generated using GemSim at 10 million reads. The distance between the mock 1 community to mock 2, mock 3, mock 4, and a replicate community (mock 1 V2) was computed using Mash, Simka (Jaccard and Bray-Curtis distance), and Libra (cosine distance, natural, and logarithmic weighting).

a stepwise increase in distance among the mock communities (Fig. 2B). This suggests that logarithmic weighting in Libra allows for a comparison of distantly related microbial communities. Mash also shows a stepwise distance between communities but is compressed relative to Libra, making differences less distinct. Simka (Bray-Curtis and Jaccard) and Libra (cosine distance, natural weighting) reach the maximum difference between mock communities 3 and 4 (Fig. 2B). This indicates that these distances are more appropriate when comparing metagenomes with small fluctuations in the community (e.g., data from a time-series

analysis), whereas Libra (cosine distance, logarithmic weighting) can be used to distinguish metagenomes that vary in both genetic composition and abundance over a wide range of species diversity by dampening the effect of high-abundance *k*-mers. Because of this important difference, we used the cosine distance with the logarithmic weighting in all subsequent analyses. Further, we also found that cosine distance provides the fastest computation among all distance metrics (see the Methods section). We confirmed these findings using Illumina simu-

lated datasets (Supplementary Fig. S1A) to show that these re-sults are consistent across short-read technologies.

Given the availability of long-read (∼10K) sequencing tech-nologies such as Oxford Nanopore and PacBio sequencing, we repeated the above analyses on simulated long-read data (Sup-plementary Fig. S1B). We show that simulated PacBio long-read data for the mock community derived from SimLoRD [42] shows a similar stepwise distance pattern between each of the mock communities (Supplementary Fig. S1B) but has a higher overall distance between mock 1 and each of the mock communities (mock 2–4), likely due to the high simulated random error rate compared to simulated short-read data.

Libra accurately profiles differences in bacterial diversity and abundance in amplicon and WGS datasets from the human microbiome

Microbial diversity is traditionally assessed using two methods: the 16S rRNA gene to classify bacterial and archaeal groups at the genus to species level or WGS for finer taxonomic classifica-tion at the species or subspecies level. Further, WGS datasets provide additional information on functional differences be-tween metagenomes. Here, we compare and contrast the effect of different algorithmic approaches (Mash vs Libra vs Simka), distance metric (Libra vs Simka), data type (16S rRNA vs WGS), and sequence type (WGS reads vs assembled contigs) in analyz-ing data from 48 samples across eight body sites from the HMP. Specifically, we examine matched datasets (16S rRNA reads, WGS reads, and WGS assembled contigs) classified as urogenital (posterior fornix), gastrointestinal (stool), oral (buccal mucosa, supragingival plaque, tongue dorsum), airways (anterior nares), and skin (retroauricular crease left and right) (See Supplemen-tary Table S2).

Because the HMP datasets represent microbial communities, abundant bacteria will have more total read counts than rare bacteria in the samples. Thus, each sample can vary by both taxonomic composition (the genetic content of taxa in a sam-ple) and abundance (the relative proportion of those taxa in the samples). Importantly, the 16S rRNA amplicon dataset is useful in showing how well each algorithm performs in detecting and quantifying small-scale variation for single a gene at the genus level, whereas the WGS dataset demonstrates the effect of in-cluding the complete genetic content and abundance of organ-isms at the species level in a community [47]. Also, we examine differences in each algorithm when read abundance is excluded using assembled contigs that only represent the genetic compo-sition of the community.

Using the 16S rRNA reads, both Mash and Libra clustered samples by broad categories but not individual body sites (Fig. 3A and 3B). Similar to what has been described in previous work [15], samples from the airways and skin co-cluster, whereas other categories, including urogenital, gastrointestinal, and oral, are distinct [15]. These results indicate that limited variation in the 16S rRNA gene may only allow for clustering for broad cat-egories. Further, the Mash algorithm shows lower overall res-olution (Fig. 3A) compared to Libra (Fig. 3B). Indeed, amplicon sequencing analysis is not an original intended use of Mash, given that it reduces the dimensionality of the data by looking at presence/absence of unique k-mers, whereas Libra examines the complete dataset accounting for both the genetic composition of organisms and their abundance. In contrast, Simka (Jaccard-ab and Bray-Curtis) fails to cluster samples by broad categories; some skin samples are found associated with stool and fornix samples (Fig. 3C and 3D). Moreover, Simka Jaccard-ab fails to

cluster the mouth samples together (Fig. 3C). This result sug-gests that applying Simka and these well-used distance metrics are not appropriate for these datasets.

When using WGS reads, both Mash and Libra show enhanced clustering by body site (Fig. 4A and B); however, Mash shows decreased resolution (Fig. 4A) as compared to Libra (Fig. 4B). Again, these differences reflect the effect of using all of the read data (Libra) rather than a subset (Mash). The effect of us-ing all of the read data compared to a subset (when sketching in Mash) has been previously described in Benoit et al. [15]. Im-portantly, the Libra algorithm depends on read abundance that provides increased resolution for interpersonal variation as seen in skin samples (Fig. 4B). Similar to the 16S rRNA datasets, Simka (Jaccard-ab and Bray-Curtis) failed to cluster the samples by body site, where some skin and stool samples cluster with fornix samples (Fig. 4C and 4D). Similarly, Simka Jaccard-ab also fails to cluster the mouth samples together (Fig. 4C). Overall Simka shows an enhanced clustering by body site using WGS data com-pared to the 16S rRNA data using these distance metrics; how-ever, the clustering is still not accurate. In order to confirm the independence of these results toward the sequencing technol-ogy, we performed the same experiment on the *CAMI HMP* "toy dataset" (simulated PacBio long reads) (Supplementary Fig. S2). This analysis shows that each of the tools is able to cluster the samples broadly by body site. However, there are small misclas-sifications shared across all tools, suggesting that the increased error rate for this technology could have a limited impact on k-mer–based analytics.

When abundance is taken out of the equation by using as-sembled contigs (see Supplementary Fig. S3), Mash performs well in clustering distinct body sites, whereas Libra shows dis-crepancies and less overall resolution. Thus, as designed, Libra requires reads rather than contigs to perform accurately and obtain high-resolution clustering (Fig. 4). Simka (Jaccard-ab and Bray-Curtis) was not able to distinguish any assembled datasets and scored all sample-to-sample distances to the maximum, even considering presence-absence distance metric proposed by Simka (data not shown). This phenomenon may be explained by the normalization method used by Simka, which does not pro-vide enough data to compare the samples when normalized by the smallest number of contigs (in our dataset 69 contigs).

Libra allows for ecosystem-scale analysis: clustering theTOVs to unravel global patterns

To demonstrate the scale and performance of the Libra algo-rithm, we analyzed 43 TOVs from the 2009–2011 Expedition [36] representing 26 sites, 43 samples, and 4.2 billion reads from the global ocean (see the Methods section). Phages (viruses that in-fect bacteria) are abundant in the ocean [48] and can signif-icantly impact environmental processes through host mortal-ity, horizontal gene transfer, and host-gene expression. Yet, how phages change over space and time in the global ocean and with environmental fluxes is just beginning to be explored. The pri-mary challenge is the majority of reads in viromes (often >90%) do not match known proteins or viral genomes [3] and no con-served genes like the bacterial 16S rRNA gene exist to differen-tiate populations. To examine known and unknown viruses si-multaneously, viromes are best compared using sequence sig-natures to identify common viral populations.

Two approaches exist to cluster viromes based on sequence composition. The first approach uses protein clustering to ex-amine functional diversity in viromes between sites [3, 36, 49]. Protein clustering, however, depends on accurate assembly and
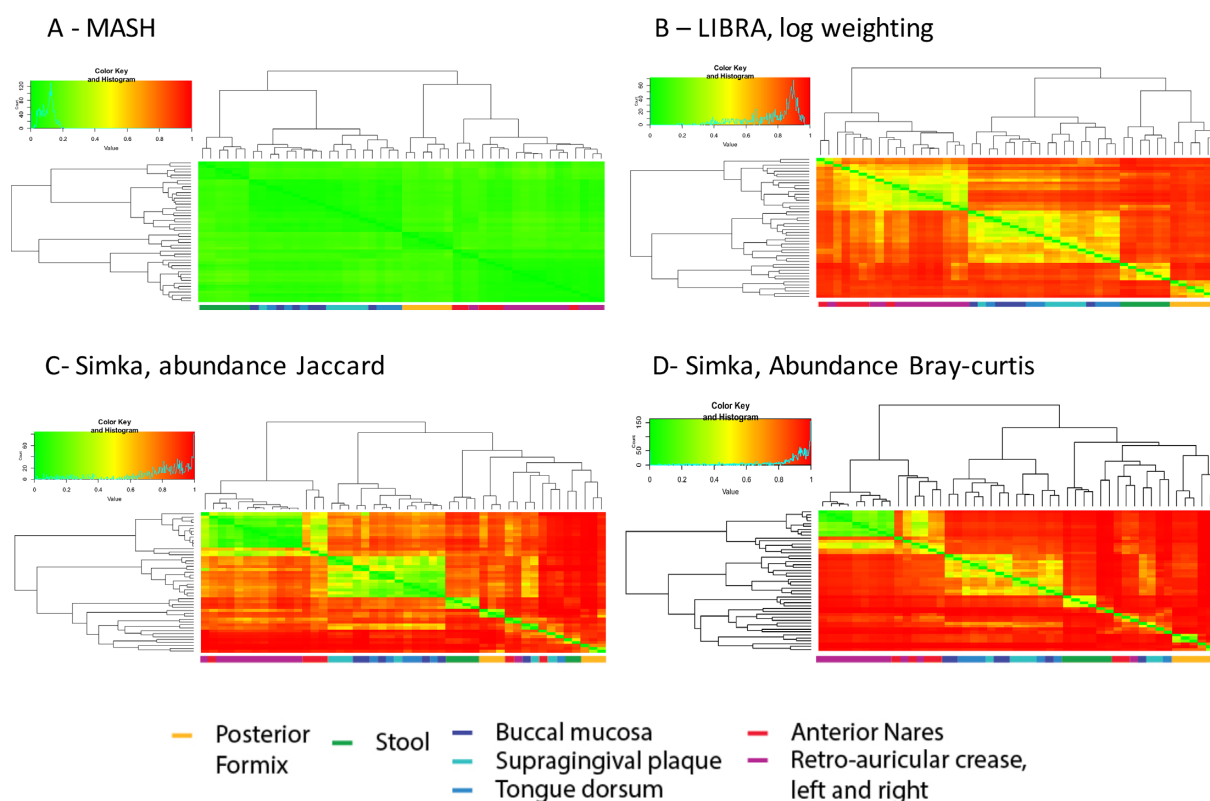
A - MASH
B – LIBRA, log weighting
C- Simka, abundance Jaccard
D- Simka, Abundance Bray-curtis

— Posterior Formix   — Stool   — Buccal mucosa   — Supragingival plaque   — Tongue dorsum   — Anterior Nares   — Retro-auricular crease, left and right

Figure 3: Clustering of HMP 16S rRNA datasets using Mash, Libra, and Simka. A total of 48 human metagenomic samples from the HMP clustered by Mash **(A)**, Libra **(B)**, or Simka using Jaccard-ab **(C)** and Bray-Curtis distances **(D)** from 16S rRNA sequencing runs. The samples were clustered using Ward's method on their distance scores. Mash, Simka, and Libra report distance in the same range (0–1). Heat maps showing the pairwise dissimilarity between samples were therefore scaled between 0 (green) and 1 (red). A key below the heat map colors the samples by body sites.

Table 1: Execution times for the Libra based on the TOV dataset

Stage	Execution time
Preprocessing (k-mer histogram construction/Inverted index construction)	16:32:55
Distance matrix computation	1:24:27
Total	17:57:22

gene finding that can be problematic in fragmented and genetically diverse viromes [50]. Further, assemblies from viromes often include only a fraction of the total reads (e.g., only one-third in TOVs [36]). To examine global viral diversity in the ocean using all of the reads, we examined TOVs using Libra. The complete pairwise analysis of ~4.2 billion reads in the TOV dataset [36] finished in 18 hours using a 10-node Hadoop cluster (see the Methods section and Supplementary Table S4). Importantly, Libra exhibits remarkable performance in computing the distance matrix, wherein k-mer matches for all TOVs completed within 1.5 hours (see Table 1). This step usually represents the largest computational bottleneck for bioinformatics tools that compute pairwise distances between sequence pairs for applications such as hierarchical sequence clustering [51–54]. A direct comparison of the runtime of Simka, Mash, and Libra are not possible given that each tool is tuned to a different computational architecture with a different number of servers and total

CPU/memory (Mash runs on a single server; Simka runs on an HPC, and Libra on Hadoop).

Overall, we found that viral populations in the ocean are largely structured by temperature in four gradients (Fig. 5) similar to their bacterial hosts [2]. Interestingly, samples from different Longhurst provinces but the same temperature gradient cluster together. Also, water samples from the surface and DCM at the same station, cluster more closely together than samples from the same depth at nearby sites (Fig. 5). Also noteworthy, samples that were derived from extremely cold environments (noted as C0 in Fig. 5) lacked similarity to all other samples (at a 30% similarity score), indicating distinctly different viral populations. These samples include a mesotrophic sample that has previously been shown to have distinctly different viral populations than surface ocean samples [55]. Taken together, these data indicate that viral populations are structured globally by temperature and at finer resolution by the station (for surface and DCM samples), indicating that micronutrients and local conditions play an important role in defining viral populations.

Innovations

Scientific collaboration is increasingly data driven given large-scale next-generation sequencing datasets. It is now possible to generate, aggregate, archive, and share datasets that are terabytes and even petabytes in size. Scalability of a system is becoming a vital feature that decides the feasibility of massive 'omic's analyses. In particular, this is important for metagenomics where patterns in global ecology can only be discerned
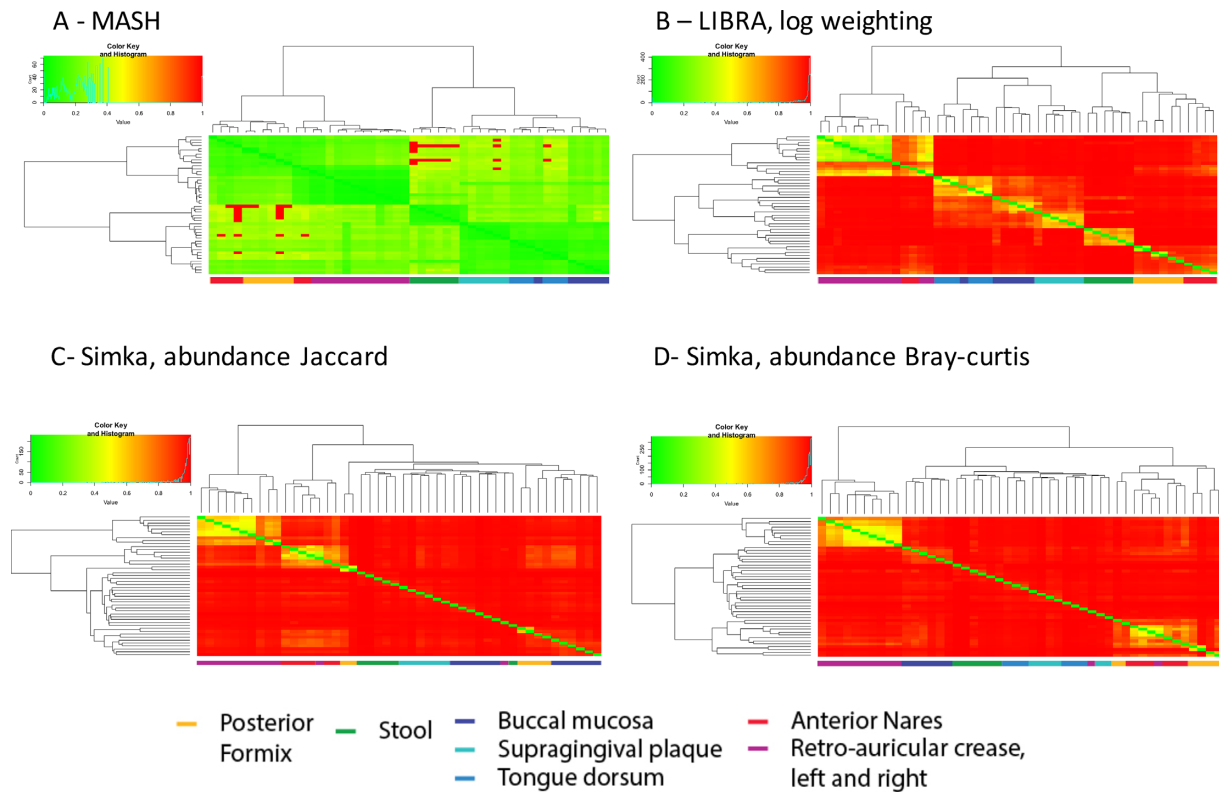
Figure 4: Clustering of WGS samples using Mash, and Libra and Simka. A total of 48 human metagenomic samples from the HMPs clustered by Mash (**A**), Libra (**B**), or Simka using Jaccard-ab (**C**) and Bray-Curtis distances (**D**) from WGS runs. The samples were clustered using Ward's method on their distance scores. Heat maps illustrate the pairwise dissimilarity between samples, scaled between 0 (green) and 1 (red). A key below the heat map colors the samples by body site.

by comparing the sequence signatures of microbial communities from massive 'omics datasets, given that most microbial genomes have not been defined. Current algorithms to perform these tasks run on local workstations or high-performance computing architectures.

Hadoop is a well-used framework allowing for scalability. The Hadoop framework was previously used for *k*-mer spectra calculation in prior work (Supplementary Table S1B) [31, 32]. However, these tools do not provide any distance computation between the generated *k*-mer spectra. To our knowledge, Libra is, therefore, the first *k*-mer based *de novo* comparative metagenomic tool that uses a Hadoop framework for scalability and fault tolerance.

De novo comparative metagenomic tools rely on the calculation of a distance metric in order to perform a clustering task on the metagenomes. Libra provides several distance metrics on the *k*-mer spectra: two well-used metrics in metagenomics (Bray-Curtis and Jensen distance), as well as a Cosine Similarity metric. Cosine Similarity, although extensively used in computer science, has been rarely implemented in genomic and metagenomic studies [46]. To our knowledge, this work is the first to describe the use of the Cosine Similarity metric to cluster metagenomes based on their *k*-mer content.

Finally, the analysis of large-scale metagenomic analysis requires access to large computing resources. In order to use Libra, the user requires access to a Hadoop framework. To allow for better access to the tool and to computing resources, we provide a web-based implementation tool embedded in the CyVerse advanced cyberinfrastructure through iMicrobe [37]. The work described here is the first step in implementing a free cloud-based computing resource for *de novo* comparative metagenomics that

can be broadly used by scientists to analyze large-scale shared data resources. Moreover, the code can be ported to any Hadoop cluster (e.g., Wrangler at TACC, Amazon EMR, or private Hadoop clusters). This computing paradigm is consistent with recent efforts to increase the accessibility of big datasets in the cloud, such as the Pan-Cancer Analyses of Whole Genomes Project [56].

Methods

Libra algorithm detailed description

k-mer size

Libra calculates the distances between samples based on their *k*-mer composition. The canonical representation of the *k*-mer is used to reduce the number of stored *k*-mers. Several considerations should be taken into account for choosing the *k*-mer size *k*. Larger values of *k* result in fewer matches due to sequencing errors and fragmentary metagenomic data. However, smaller values of *k* give less information about the sequence similarities. In Libra, *k* is a configurable parameter chosen by the user and is set by default to *k* equal to 21. This value was reported to be at the inflection point where the *k*-mer matches move from random to a representative of the read content and is generally resilient to sequencing error and variation [57, 58].

Distance matrix computation

Libra provides three distance metrics—Cosine Similarity, Bray-Curtis, and Jensen-Shannon. Cosine Similarity is the default.
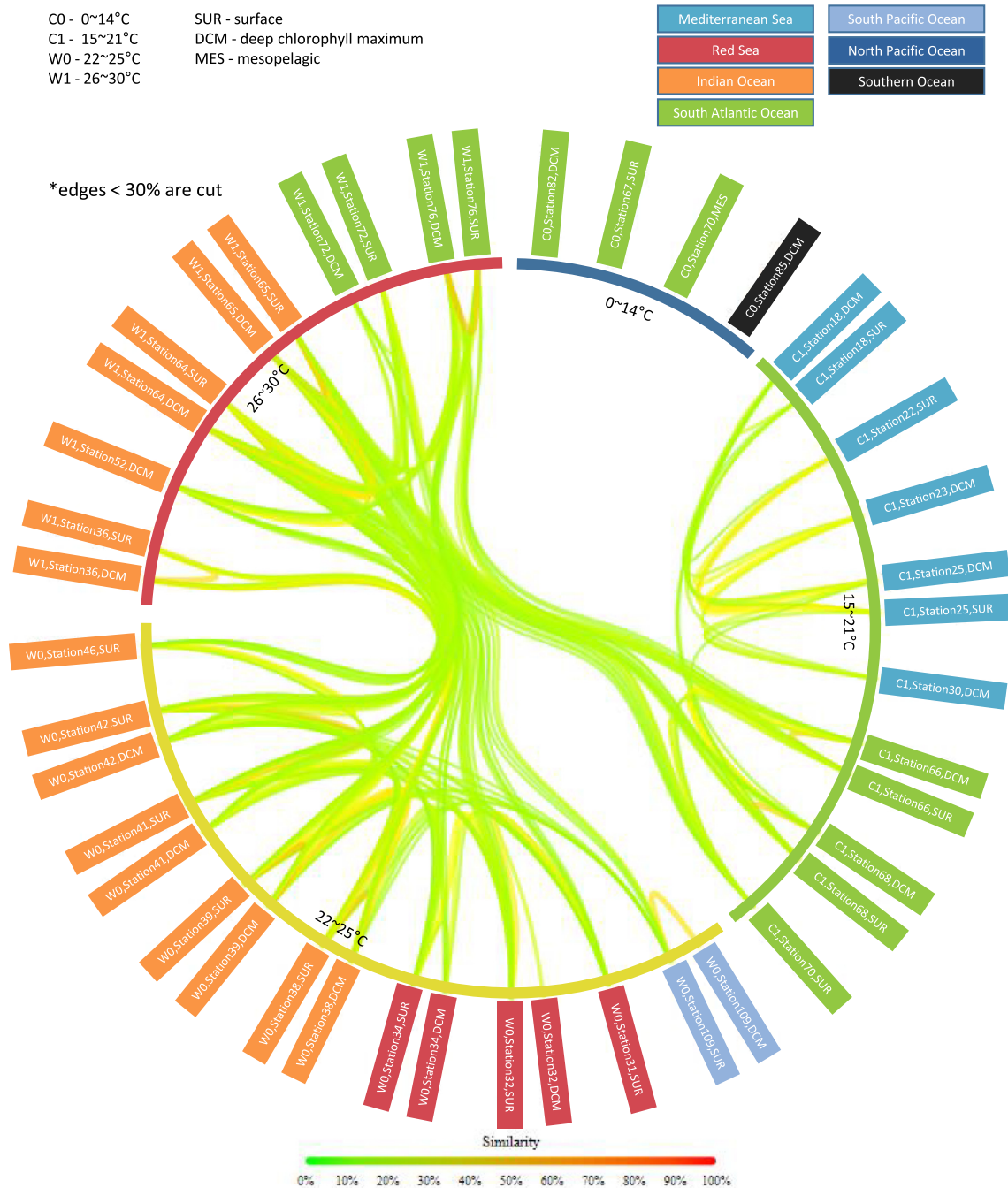
Figure 5: Visualizing the genetic distance among marine viral communities using Libra. Similarities between samples from 43 TOV from the 2009–2012 Tara Oceans Expedition. Lines (edges) between samples represent the similarity and are colored and thickened accordingly. Lines with insignificant similarity (less than 30%) are removed. Each of the sample names is color coded by Longhurst Province. Inner circles show temperature ranges. Sample names show the temperature range, station, and depth as indicated on the legend. The analysis is performed using Libra (k = 20, Logarithmic weighting, and Cosine Similarity).

Cosine Similarity metric

Libra constructs a vector v_s for each sample s from the weight of each k-mer k in the sample ($w_{k,s}$). Each dimension in the vector corresponds to the weight of the corresponding k-mer:

$$v_s = (w_{k1,s}, w_{k2,s}, w_{k3,s}, \ldots, w_{kn,s})$$

The weight of a k-mer in a sample (w_{ks}) can be derived from the frequency of the k-mer (f_{ks}) in several ways. The simplest

uses the raw frequency of the k-mer ($w_{ks} = f_{ks}$), called *Natural Weighting*. Another uses *Logarithmic Weighting* ($w_{ks} = 1 + log(f_{ks})$) to not give too much weight to highly abundant k-mers. In this weighting, w_{ks} grows logarithmically with the frequency f_{ks}, reducing the effect on the distance of highly abundant k-mers caused by sequencing artifacts.

Once their vectors have been constructed, the distance between two samples (s_1 and s_2) is derived using distance metrics. For example, the distance between the two samples using Co-

sine Similarity is determined as follows:

$$Distance\,(s_1, s_2) \;=\; 1 - CosineSimilarity\,(s_1, s_2)$$

$$= 1 - \cos\,(v_{s1}, v_{s2}) \;=\; 1 - \frac{v_{s1} \,\cdot\, v_{s2}}{||v_{s1}|| \,\times\, ||v_{s2}||} \;=\; 1 - \frac{D_{s1,s2}}{M_{s1} \times M_{s2}}$$

where,

$$D_{s1,s2} = v_{s1} \,\cdot\, v_{s2} = \sum_{i \in s1 \cap s2} w_{ki,s1} \times w_{ki,s2},$$

$$M_s = ||v_s|| = \sqrt{\sum_{i \in s} \left(w_{ki,s}\right)^2}$$

In other words, $D_{s1,\,s2}$ is the dot product of the vectors v_{s1} and v_{s2}, and M_s is the magnitude (length) of the vector v_s. The distance between two NGS samples is the cosine of the angle between their vectors v_s; the magnitude of the vector M_s is not taken into account in the metric, thereby normalizing samples with different numbers of total base pairs.

Inverted index construction

A naïve implementation would require the storage of one vector with 4^k dimensions per sample, where k is the k-mer length. For a k of 21, each vector would have more than 1 million dimensions. To reduce the overhead, Libra stores and computes the distance on a single *inverted index* with the k-mer frequencies from multiple samples and performs the distance computation on the index directly. The inverted index is indexed by k-mer, and each entry is an index record containing a list of pairs, each of which contains a sample identifier and the frequency of the k-mer in the sample.

$$index\,record = k - mer : \{\langle sample - id, frequency \rangle,$$
$$\langle sample - id, frequency \rangle \ldots \}$$

The records in the index are stored in alphabetical order by k-mer, allowing the record for a particular k-mer to be found via binary search. The k-mer record contains the k-mer frequency in each sample, not the weight, to allow for different weighting functions to be applied during distance matrix computation.

Sweep line algorithm

To compute the distance between two samples S_1 and S_2, Libra must compute the three values $D_{s1,\,s2}$, M_{s1}, and M_{s2}. The values are calculated by scanning through the vectors v_{s1} and v_{s2} and computing the values. The time for the distance matrix computation is proportional to the number of dimensions (the number of k-mers) in the two vectors. In general, computing all-vs-all comparisons on n samples would require $n \times (n-1)/2$ vector scans, which becomes prohibitively expensive as n gets large. Libra uses a sweep line algorithm [38] to greatly reduce the computational time. The sweep line algorithm only requires a single scan of all vectors to compute the distance of all pairs of samples (see Supplementary Fig. S4). Briefly, Libra sweeps a line through all the vectors simultaneously starting with the first component.

Libra outputs a record of the non-zero values of the following format:

$$record = k - mer : \{\langle sample - id, weight \rangle, \langle sample - id, weight \rangle, \ldots \}$$

Libra then moves the sweep line to the next component and performs the same operation. From the output records, contributions to M_s for each sample in the record are computed and accumulated. Contributions to D are also computed from the record by extracting sample pairs. For example, the record $\{< s_1, \; x >, \; < s_2, \; y >, \; \langle s_4, \; z \rangle \}$ has three sample pairs $(s_1 s_2)$, $(s_1 s_4)$ *and* $(s_2 s_4)$. Libra then computes contribution to D for each pair, e.g., $x * y$ is added to $D_{s1,\,s2}$, $x * z$ is added to $D_{s1,\,s4}$, and $y * z$ is added to $D_{s2,\,s4}$. Using this method, Libra computes the distances of every sample pair in an input dataset in linear time. Other distance metrics, such as Bray-Curtis and Jensen-Shannon, can also be computed in the same fashion.

The sweep algorithm is particularly easy to implement on an inverted index; it consists of simply stepping through the (sorted) k-mers. Furthermore, the sweep algorithm is easily parallelized. The k-mer space is partitioned and a separate sweep is performed on each partition computing the contributions of its k-mer frequencies to the D and M values. At the end of the computation, the intermediate D and M values are combined together to produce the final D and M values and thereby the distance matrix. Each sweep uses binary search to find the first k-mer in the partition.

Terabyte sort

Libra groups the samples automatically based on the number and size (by default 4 GB per group). Similar to Terabyte Sort [59], the index records are partitioned by k-mer ranges and the records in each partition are stored in a separate *chunk file*. All k-mers in partition n appear before the k-mers in partition $n + 1$ in lexicographic order. This facilitates breaking computation and I/O down into smaller tasks, so that work of creating an index can be distributed across several machines.

k-mer space partitioning

Both the inverted index construction and the distance matrix computation require partitioning the k-mer space so that different partitions can be processed independently. For the partitioning to be effective, the workload should be balanced across the partitions. Simply partitioning into fixed-size partitions based on the k-mer space will not ensure balanced workloads, as the k-mers do not appear with uniform frequency. Some partitions may have more k-mer records than others and thereby incur higher processing costs. Instead, the partitions should be created based on the k-mer distribution, so that each partition has roughly the same number of records (see Supplementary Fig. S5).

Computing the exact k-mer distribution across all the samples is too expensive in both space and time; therefore, Libra approximates the distribution instead. A histogram is constructed using the first six letters of the k-mers in each sample, which requires much less space and time to compute. In practice, partitioning based on this histogram adequately partitions the k-mer space so that the workloads are sufficiently balanced across the partitions.
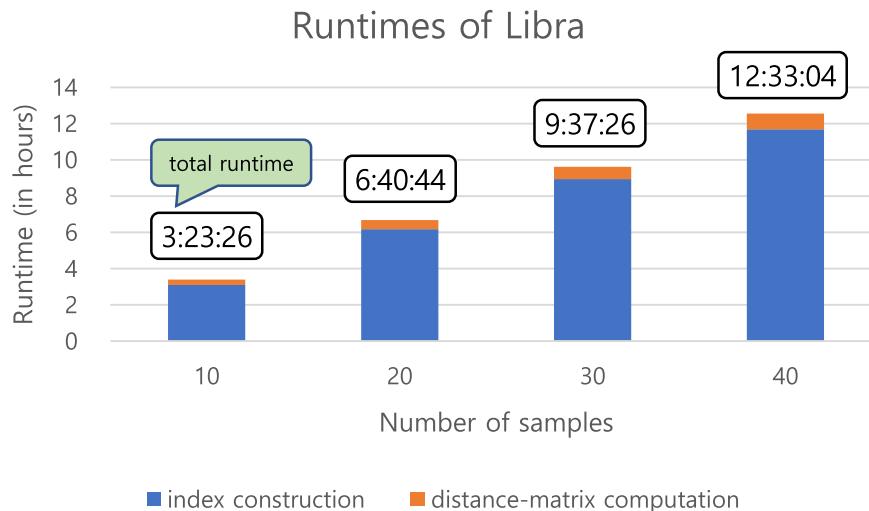
Runtimes of Libra

Figure 6: Scalability testing for Libra. Runtimes of Libra on four datasets consisting of 10, 20, 30, and 40 samples (total sizes of 93 GB, 186 GB, 279 GB, and 372 GB, respectively). Libra was performed with default parameters ($k = 20$, Logarithmic weighting, and Cosine Similarity). Runtimes were averaged out over three runs. The total runtime of Libra increased linearly with increased input volume. Both index construction and distance matrix computation showed linearly increased runtimes for the increased input volume. This shows that Libra performs efficiently and scales to input although the number of distances between sample pairs to be computed increases quadratically.

Scalability benchmarking for Libra

We used synthetic datasets for a scalability benchmark. Each dataset contains 10 billion bytes (approximately 9.3 GB). We used four datasets consisting of 10 (93 GB), 20 (186 GB), 30 (279 GB), and 40 (372 GB) samples in the benchmark. Each experiment was run three times, and an average of the three runs reported (see Supplementary Table S4 for details). The runtime of Libra increased linearly with increased input volume (Fig. 6). This shows that Libra efficiently handles the increased volume of input and efficiently computes distances between all sample pairs while the number of sample pairs increases quadratically.

Benchmarking runtimes of different distance metrics in Libra

We used the same synthetic dataset with 40 samples (372 GB in total) in the scalability benchmarking (Fig. 7). We measured the runtimes of Libra for the different distance metrics. Once the index is constructed, all distance metrics are calculated using that index; thus, runtimes of the inverted index construction for the different metrics are the same. Each experiment was run three times and the average reported (see Supplementary Table S4 for details). Differences in runtimes are mainly due to the different computational workloads of distance metrics (Fig. 7). For example, Jensen-Shannon requires more multiplications and divisions in nested loops than Cosine Similarity, incurring more computational workload. Yet, distance matrix computation with Jensen-Shannon took only 12.64% of total runtime.

Advanced cyberinfrastructure for Libra in iMicrobe

To improve access to Libra, we made it available on the iMicrobe website [37]. A researcher with a CyVerse account can run Libra on iMicrobe by filling out a simple web form specifying the input files and parameters. Input files are selected from the CyVerse Data Store where they have either been uploaded by the user to their home directory or are part of the iMicrobe Data Commons. When a job is submitted, the user is presented with the status

of the job and on completion with the output files and visualization of results. To deploy Libra on iMicrobe, we developed a job dispatch service to automate the execution of Libra on a University of Arizona Hadoop cluster. The service is written in NodeJS and accepts a JSON description of the job inputs and parameters, stages the input files onto the UA Hadoop cluster, executes Libra with the given parameters, and transfers the resulting output files to the user's home directory in the CyVerse Data Store. The service provides a RESTful interface that mimics the Agave API Jobs service and is secured using an Agave OAuth2 token. The source code is available on Github [60].

Experimental environment description

Mash and Simka configurations
Mash v1.1 was run on the metagenomic datasets with the following parameters: -r –s 10000 –m 2 [19]. The analysis of assemblies was run without the parameter "-r," used for short sequences.

Simka v1.3.2 was run on the metagenomic datasets with the following parameters: -abundance-min 2 -max-reads [MIN-COUNT] -simple-dist -complex-dist, where [MINCOUNT] is the smallest sequence count across the analyzed samples.

Hadoop cluster configuration
The Libra experiments described here were performed on a Hadoop cluster consisting of 10 physical nodes (9 MapReduce worker nodes). Each node contains 12 CPUs and 128 GB of RAM and is configured to run a maximum of 7 YARN containers simultaneously with 10 GB of RAM per container. The remaining system resources are reserved for the operating system and other Hadoop services such as Hive or HBase.

The rationale for not porting Libra to Spark
Spark [61] is increasingly popular for scientific data analysis [62] because of its outstanding performance provided by fast in-memory processing. Although Libra is currently implemented on Hadoop MapReduce, Libra can be easily ported to Spark because both Hadoop MapReduce and Spark have similar inter-
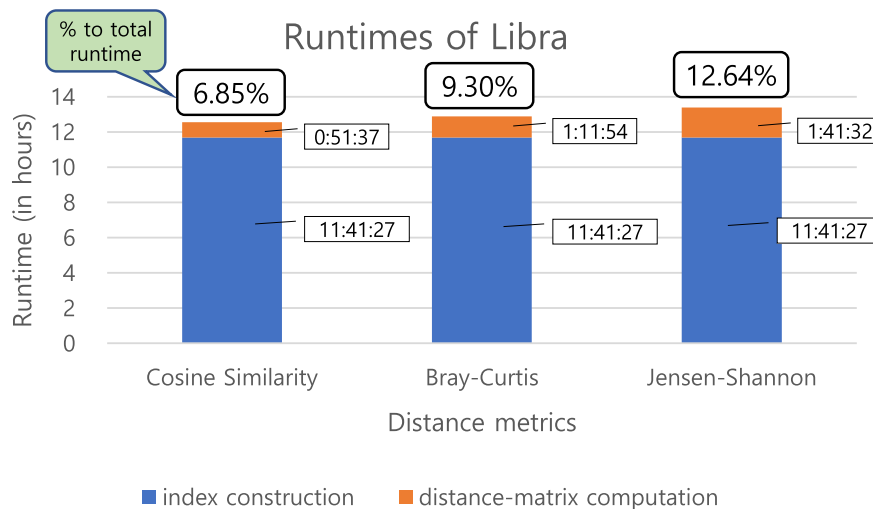
Figure 7: Runtimes for different distance metrics. Runtimes for three different distance metrics (Cosine Similarity, Bray-Curtis, and Jensen-Shannon) in Libra with 40 samples of input (372 GB in total). Libra was performed with default parameters ($k = 20$ and Logarithmic weighting). Runtimes were averaged over three runs. An inverted index was reused for all three distance metrics because the inverted index Libra constructs are independent of the distance metrics. Cosine Similarity took the shortest runtime among the three metrics while Jensen-Shannon took the longest. Jensen-Shannon took almost twice as long as Cosine Similarity because it requires more mathematical computations. Because of its fastest runtime, Cosine Similarity is used by default in Libra.

faces for data processing and partitioning. For example, resilient distributed datasets (RDD) can be partitioned and distributed over a Spark cluster using Libra's k-mer range partitioning. RDDs are memory resident, allowing Spark to significantly improve the performance of Libra's k-mer counting and distance matrix computation by avoiding slow disk I/O for intermediate data. We implemented Libra using Hadoop MapReduce because Spark requires much more RAM than Hadoop MapReduce, significantly increasing the cost of the cluster.

Availability of source code and requirements

Project home page: Program binary, source code, and documentation for Libra are available in Github [63]; Libra web-based App is in iMicrobe [37] under Apps; code to implement the Libra web-based App is in Github [60].

    Operating system(s): MapReduce 2.0 (Apache Hadoop 2.3.0 or above).

    Programming language: Java 7 (or above).

    Other requirements: None.

    License: Apache license version 2.0.

    Any restrictions to use by non-academics: None. Libra has been registered with the SciCrunch database under reference ID: SCR_016608.

Availability of supporting data

Snapshots of the code and other supporting data are available in the *GigaScience* repository, GigaDB [64].

Additional files

supplemental_Fig1.pdf
Supplemental Fig2.pdf
Supplemental Fig3.pdf
Supplemental Fig4.pdf
Supplemental Fig5.pdf
Supplemental Table1 -Comparable_tools.xlsx
Supplemental Table 2.xlsx
Supplemental Table 3.xlsx
Supplemental Table 4.xlsx
Supplemental Table 5.xlsx

Abbreviations

CAMI: Critical Assessment of Metagenome Interpretation; DCM: deep chlorophyll maximum; HDFS: high-performance distributed file system; HMP: Human Microbiome Project; HPC: high-performance computing cluster; NGS: next-generation sequence; PacBio: Pacific Biosciences; RDD: resilient distributed dataset; TOV: Tara Ocean Virome; WGS: whole-genome shotgun sequencing.

Competing interests

The authors declare that they have no competing interests.

Funding

Author contributions

B.L.H., J.H.H., and I.C. conceived of the Libra algorithm and code. I.C. wrote the Libra code. B.L.H., A.J.P., and I.C. planned and carried out the analyses. B.L.H., A.J.P., J.H.H., and I.C. contributed to the interpretation of the results. M.B. and K.Y.C. implemented Libra as an application in iMicrobe. B.L.H., A.J.P., J.H.H., and I.C. wrote the manuscript. All authors provided critical feedback and helped shape the research, analysis, and manuscript.

Acknowledgments

References

1. Yooseph S, Sutton G, Rusch DB, et al. The Sorcerer II Global Ocean Sampling expedition: expanding the universe of protein families. PLoS Biol 2007;**5**:e16.

2. Sunagawa S, , Coelho LP, Chaffron S, et al. Structure and function of the global ocean microbiome. Science . 2015;**348**, 1261359. http://www.sciencemag.org/content/348/6237/1261359.abstract.

3. Hurwitz BL, Sullivan MB. The Pacific Ocean Virome (POV): a marine viral metagenomic dataset and associated protein clusters for quantitative viral ecology. PLoS One 2013;**8**:e57355.

4. Marcy Y, Ouverney C, Bik EM et al. Dissecting biological "dark matter" with single-cell genetic analysis of rare and uncultivated TM7 microbes from the human mouth. Proc Natl Acad Sci U S A. 2007;**104**:11889–94.

5. Rinke C, Schwientek P, Sczyrba A, et al. Insights into the phylogeny and coding potential of microbial dark matter. Nature. 2013;**499**:431–7.

6. Hug LA, Baker BJ, Anantharaman K, et al. A new view of the tree of life. Nat Microbiol.2016;**1**:16048.

7. Dubinkina VB, Ischenko DS, Ulyantsev VI, et al. Assessment of k-mer spectrum applicability for metagenomic dissimilarity analysis. BMC Bioinformatics.2016;**17**:38.

8. Teeling H, Waldmann J, Lombardot T, et al. TETRA: a webservice and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in DNA sequences. BMC Bioinformatics 2004;**5**:163.

9. Wu Y-W, Ye Y. A novel abundance-based algorithm for binning metagenomic sequences using l-tuples. J. Comput. Biol. 2011;**18**:523–34.

10. Fofanov Y, Luo Y, Katili C, et al. How independent are the appearances of n-mers in different genomes? Bioinformatics.2004;**20**:2421–8.

11. Maillet N, Lemaitre C, Chikhi R, et al. Compareads: comparing huge metagenomic experiments. BMC Bioinformatics. 2012;**13**(Suppl 19):S10.

12. Maillet N, Collet G, Vannier T, et al. Commet: comparing and combining multiple metagenomic datasets. 2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). 2014. p. 94–8.

13. Ulyantsev VI, Kazakov SV, Dubinkina VB, et al. MetaFast: fast reference-free graph-based comparison of shotgun metagenomic data. Bioinformatics. 2016;**32**:2760–7.

14. Ondov BD, Treangen TJ, Melsted P, et al. Mash: fast genome and metagenome distance estimation using MinHash. Genome Biol.2016;**17**:132.

15. Benoit G, Peterlongo P, Mariadassou M, et al. Multiple comparative metagenomics using multiset k-mer counting. PeerJ Comput Sci.2016;**2**:e94.

16. Broder AZ. On the resemblance and containment of documents. Proceedings Compression and Complexity of Sequences 1997 (Cat No97TB100171). http://dx.doi.org/10.1109/sequen.1997.666900

17. Koslicki D, Zabeti H. Improving Min Hash via the containment index with applications to metagenomic analysis . bioRxiv 2017. 184150, doi:10.1101/184150. https://www.biorxiv.org/content/early/2017/09/04/184150.abstract.

18. Brown CT, Irber L. sourmash: a library for MinHash sketching of DNA. The Journal of Open Source Software , 2016;**1**, 5, 27, doi:10.21105/joss.0027. http://www.theoj.org/joss-papers/joss.00027/10.21105.joss.00027.pdf.

19. Seth S, Välimäki N, Kaski S, et al. Exploration and retrieval of whole-metagenome sequencing samples. Bioinformatics.2014;**30**:2471–9.

20. Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Commun. ACM. New York, NY, USA: ACM; 2008;**51**:107–13.

21. Kolker N, Higdon R, Broomall W, et al. Classifying proteins into functional groups based on all-versus-all BLAST of 10 million proteins. OMICS.2011;**15**:513–21.

22. McKenna A, Hanna M, Banks E, et al. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome Res.2010;**20**:1297–303.

23. Langmead B, Schatz MC, Lin J, et al. Searching for SNPs with cloud computing. Genome Biol.2009;**10**:R134.

24. Wall DP, Kudtarkar P, Fusaro VA, et al. Cloud computing for comparative genomics. BMC Bioinformatics 2010;**11**:259.

25. Langmead B, Hansen KD, Leek JT. Cloud-scale RNA-sequencing differential expression analysis with Myrna. Genome Biol.2010;**11**:R83.

26. Jourdren L, Bernard M, Dillies M-A, et al. Eoulsan: a cloud computing-based framework facilitating high throughput sequencing analyses. Bioinformatics.2012;**28**:1542–3.

27. Nguyen T, Shi W, Ruden D. CloudAligner: a fast and full-featured MapReduce based tool for sequence mapping. BMC Res Notes.2011;**4**:171.

28. Schatz MC. BlastReduce: high performance short read mapping with MapReduce. University of Maryland, ; 2008; https://www.cs.umd.edu/sites/default/files/scholarly_papers/MichaelSchatz_1.pdf

29. Schatz MC. CloudBurst: highly sensitive read mapping with MapReduce. Bioinformatics. Oxford University Press; 2009;**25**:1363–9.

30. Pandey RV, Schlötterer C. DistMap: a toolkit for distributed short read mapping on a Hadoop cluster. PLoS One. 2013;**8**:e72614.

31. Nordberg H, Bhatia K, Wang K, et al. BioPig: a Hadoop-based analytic toolkit for large-scale sequence data. Bioinformatics. 2013;**29**:3014–9.

32. Gao T, Guo Y, Wei Y, et al. Bloomfish: a highly scalable distributed K-mer counting framework. , 2017 IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS). 2017. p. 170–9.

33. Menon RK, Bhat GP, Schatz MC. Rapid parallel genome indexing with MapReduce. Proceedings of the Second International Workshop on MapReduce and Its Applications. New York, NY, USA: ACM; 2011. p. 51–8.

34. Michie MG. Use of the Bray-Curtis similarity measure in cluster analysis of foraminiferal data. Math. Geol. 1982;**14**:661–7.

35. Lin J. Divergence measures based on the Shannon entropy. IEEE Trans. Inf. Theory 1991;**37**:145–51.

36. Brum JR, Ignacio-Espinoza JC, Roux S, et al. Patterns and ecological drivers of ocean viral communities. Science; 2015;**348**:1261498. http://www.sciencemag.org/content/348/6237/1261498.abstract

37. Youens-Clark K, Bomhoff M, Hurwitz BL. iMicrobe . 2018 http://imicrobe.us

38. Goff SA, Vaughn M, McKay S, et al. The iPlant Collaborative: cyberinfrastructure for plant biology. Front Plant Sci. 2011;**2**:34.

39. Devisetty UK, Kennedy K, Sarando P, et al. Bringing your tools to CyVerse Discovery Environment using Docker. F1000Res 2016;**5**:1442.

40. Ponsero A, Hurwitz BL. Simulated metagenomics mock communities . Cyverse Data Commons. 2018: , https://doi.org/10.7946/MQ0G.

41. McElroy KE, Luciani F, Thomas T. GemSIM: general, error-model based simulator of next-generation sequencing data. BMC Genomics.2012;**13**:74.

42. Stöcker BK, Köster J, Rahmann S. SimLoRD: simulation of long read data. Bioinformatics.2016;**32**:2704–6.

43. Human Microbiome Project Consortium. Structure, function and diversity of the healthy human microbiome. Nature.2012;**486**:207–14.

44. Diepenbroek M, Grobe H, Reinke M, et al. PANGAEA—an information system for environmental sciences. Comput. Geosci.; 2002;**28**:1201–10.

45. McHardy A, Sczyrba A. Critical assessment of metagenome Iiterpretation (CAMI) 2nd Challenge . 2018 CAMI . https://data.cami-challenge.org/participate.

46. Okuda S, Tsuchiya Y, Kiriyama C, et al. Virtual metagenome reconstruction from 16S rRNA gene sequences. Nat Commun.2012;**3**:1203.

47. Watts GS, Youens-Clark K, Slepian MJ, et al. 16S rRNA gene sequencing on a benchtop sequencer: accuracy for identification of clinically important bacteria. J. Appl. Microbiol. 2017;**123**:1584–96.

48. Bergh O, Borsheim KY, Bratbak G, et al. High abundance of viruses found in aquatic environments. Nature 1989;**340**:467–8.

49. Hurwitz BL, Brum JR, Sullivan MB. Depth-stratified functional and taxonomic niche specialization in the "core" and "flexible" Pacific Ocean virome. ISME J; 2014, **9**, 2, 472–84, doi:10.1038/ismej.2014.143; http://www.nature.com/ismej/journal/vaop/ncurrent/full/ismej2014143a.html.

50. Minot S, Wu GD, Lewis JD, et al. Conservation of gene cassettes among diverse viruses of the human gut. PLoS One 2012;**7**:e42342.

51. Sun Y, Cai Y, Huse SM, et al. A large-scale benchmark study of existing algorithms for taxonomy-independent microbial community analysis. Brief Bioinform; 2012;**13**:107–21.

52. Edgar RC. Search and clustering orders of magnitude faster than BLAST. Bioinformatics 2010;**26**:2460–1.

53. Niu BF, Fu LM, Sun SL, et al. Artificial and natural duplicates in pyrosequencing reads of metagenomic data. BMC Bioinformatics 2010;**11**:187.

54. Cai Y, Sun Y. ESPRIT-Tree: hierarchical clustering analysis of millions of 16S rRNA pyrosequences in quasilinear computational time. Nucleic Acids Res, 2011;**39**:e95.

55. Hurwitz BL, Brum JR, Sullivan MB. Depth-stratified functional and taxonomic niche specialization in the "core" and "flexible" Pacific Ocean virome. ISME J 2015;**9**:472–84.

56. Weinstein JN, Collisson EA, Mills GB, et al. The cancer genome atlas pan-cancer analysis project. Nat Genet. 2013;**45**:1113–20.

57. Hurwitz BL, Westveld AH, Brum JR, et al. Modeling ecological drivers in marine viral communities using comparative metagenomics and network analyses. Proc Natl Acad Sci 2014;**111**:10714–9.

58. Kurtz S, Narechania A, Stein JC, et al. A new method to compute K-mer frequencies and its application to annotate large repetitive plant genomes. BMC Genomics 2008;**9**:517.

59. O'Malley O. Terabyte sort on Apache Hadoop. http://sortbenchmark org/Yahoo-Hadoop pdf, Citeseer; 2008;1–3.

60. Bomhoff M. Ocean Cloud Commons Plan B Github . 2018 https://github.com/hurwitzlab/occ-plan-b

61. Zaharia M, Chowdhury M, Franklin MJ, et al. Spark: cluster computing with working sets. HotCloud. 2010;**10**:95.

62. Guo R, Zhao Y, Zou Q, et al. Bioinformatics applications on Apache Spark. GigaScience . 2018, **7**, 8, doi:10.1093/gigascience/giy098.

63. Choi I. Libra Github . 2018 https://www.github.com/iychoi/Libra

64. Choi I, Ponsero AJ, Bomhoff M, et al. Supporting data for "Libra: scalable *k*-mer based tool for massive all-vs-all metagenome comparisons." GigaScience Database 2018. http://dx.doi.org/10.5524/100547.