BIOINFORMATICS ORIGINAL PAPER

Vol. 29 no. 19 2013, pages 2395–2401
doi:10.1093/bioinformatics/btt420

Sequence analysis

Advance Access publication August 5, 2013

Distilled single-cell genome sequencing and *de novo* assembly for sparse microbial communities

Zeinab Taghavi[1,*], Narjes S. Movahedi[1], Sorin Drăghici[1,2] and Hamidreza Chitsaz[1]

[1]Department of Computer Science and [2]Department of Obstetrics and Gynecology, Wayne State University, Detroit, MI 48202, USA

Associate Editor: Gunnar Ratsch

ABSTRACT

Motivation: Identification of every single genome present in a microbial sample is an important and challenging task with crucial applications. It is challenging because there are typically millions of cells in a microbial sample, the vast majority of which elude cultivation. The most accurate method to date is exhaustive single-cell sequencing using multiple displacement amplification, which is simply intractable for a large number of cells. However, there is hope for breaking this barrier, as the number of different cell types with distinct genome sequences is usually much smaller than the number of cells.

Results: Here, we present a novel divide and conquer method to sequence and *de novo* assemble all distinct genomes present in a microbial sample with a sequencing cost and computational complexity proportional to the number of genome types, rather than the number of cells. The method is implemented in a tool called Squeezambler. We evaluated Squeezambler on simulated data. The proposed divide and conquer method successfully reduces the cost of sequencing in comparison with the naïve exhaustive approach.

Availability: Squeezambler and datasets are available at http://compbio.cs.wayne.edu/software/squeezambler/.

Contact: ztaghavi@wayne.edu

Received on March 19, 2013; revised on May 22, 2013; accepted on July 15, 2013

1 INTRODUCTION

Critical applications, including the Human Microbiome Project (Methe *et al.*, 2012), biothreat detection and combating antibiotic resistant pathogens, necessitate identification of all distinct genome sequences in a bacterial sample. When prior knowledge is available about which organisms may be present in the sample, flow cytometry and 16S rRNA gene sequencing may be used. However, metagenomics is usually used for analyzing the genomics of relatively abundant microbes when no prior knowledge is given. Metagenomics consists the study of the variation of species in a complex microbial sample. Because the vast majority of environmental bacteria elude cultivation, metagenomics investigates microbial communities by sequencing sampled genome fragments without the need for culturing. Such a heterogeneous pool of sequencing reads can also be assembled to yield a superposition of highly abundant genomes in the sample (Treangen *et al.*, 2013). There are two problems with metagenomics: (i) the resulting assembly contains multiple genomes superimposed, and (ii) only highly abundant genomes survive the sampling process.

Advances in DNA amplification technology have enabled whole genome sequencing directly from individual cells without requiring growth in culture. Single-cell sequencing methods have enabled investigation of novel uncultured microbes (Kvist *et al.*, 2007; Mussmann *et al.*, 2007). These culture-independent single-cell studies are a powerful alternative to metagenomics studies. Genomic sequencing from single bacterial genomes was first demonstrated with cells isolated by flow cytometry (Raghunathan *et al.*, 2005), using multiple displacement amplification (MDA) (Dean *et al.*, 2001, 2002; Hosono *et al.*, 2003). MDA is now the preferred method for whole genome amplification from single cells (Ishoey *et al.*, 2008; Lasken, 2007). The first attempt to assemble a complete bacterial genome from one cell (Zhang *et al.*, 2006) further explored the challenges of assembly from amplified DNA, including amplification bias and chimeric DNA rearrangements. Amplification bias results in orders of magnitude difference in coverage (Raghunathan *et al.*, 2005) and absence of coverage in some regions. Chimera formation occurs during the DNA branching process by which the ϕ_{29} DNA polymerase generates DNA amplification in MDA (Lasken and Stockwell, 2007). Subsequent studies continued to improve single-cell assemblies (Hongoh *et al.*, 2008; Marcy *et al.*, 2007; Podar *et al.*, 2007; Rodrigue *et al.*, 2009; Woyke *et al.*, 2009). A nearly full potential of single-cell genome assembly has recently been realized by the work of Chitsaz *et al.* (2011) followed by those of Bankevich *et al.* (2012) and Peng *et al.* (2012).

Owing to recent progress in single-cell DNA amplification techniques and *de novo* assembly algorithms, the genomes of all bacterial species in a sample can be captured one cell at a time. However, there are often millions of cells per sample, in which case the naïve deep sequencing of every cell becomes prohibitively costly. Moreover, it is expected that deep sequencing of every cell is often not necessary, as the majority of biologically important samples are sparse in the sense that many cells are biological replicates. Compressed (Candès and Tao, 2005, 2006; Donoho, 2006) and distilled (adaptive sampling and refinement) sensing methods (Haupt *et al.*, 2011; Wei and Hero, 2012) have been proposed in the past decade to exploit sparsity for reducing the cost of sensing and reconstructing signals in various spaces, ranging from Banach spaces to Boolean algebras (Erlich *et al.*, 2010; Stobbe and Krause, 2012). Inspired by those advances, we give an

*To whom correspondence should be addressed.

algorithm in this article that exploits sample sparsity to reduce the cost of sequencing without compromising the accuracy of identification of all distinct genomes, even the ones that are minimally represented in the sample.

2 APPROACH

A naïve approach to solve the problem, which we call *single-cell co-assembly strategy*, is to amplify the genome of each cell, barcode them individually, pool them, sequence in one sequencing run and demultiplex based on the barcode. In this approach, each cell should be isolated and its DNA extracted and amplified using MDA. Although there is currently no high-throughput device to perform these processes, one could envision automated microfluidic devices that will be capable of high-throughput separation, DNA extraction, amplification and barcoding of single cells in the near future. The output sequencing reads could then be co-assembled using our tool HyDA (Movahedi *et al*., 2012). In HyDA, the read dataset of each cell is assigned a unique color. All the colors are co-assembled in one colored de Bruijn graph. This approach requires enough unique barcodes to tag the fragments of each cell. Also, barcodes attached to each fragment are sequenced, which imposes additional sequencing cost. Fabrication of so many unique barcodes becomes prohibitively expensive for a large number of cells, and therefore this naïve approach will not work.

The number of distinct genomes in a microbial community is often much less than the number of cells. For example, Qin *et al*. (2010) estimated the number of detected distinct species in the human gut to be in the order of 1000, whereas the number of microbial cells in a human body, most of which reside in the gut, is in the order of 100 trillion. We call this effect the *sparsity* of distinct genomes in a sizable microbial population. The naïve approach does not exploit the sparsity to reduce the cost of sequencing. Here, we proposed to exploit the spareness by adopting a divide and conquer strategy to reduce the amount of required barcodes and sequencing. After isolation of each cell and extraction of the DNA, every DNA is amplified and stored separately, e.g. in a microfluidic droplet. The main idea is to sample the amplified DNAs adaptively, which is essentially allocating sequencing and barcoding resources dynamically over the course of multiple sensing iterations. Initially, the algorithm has one group of cells, which is the entire sample. In each iteration, each group is divided into two equally sized subgroups. A small amount of DNA from each cell in a subgroup is sampled, pooled and sequenced. In practice, one barcode per subgroup is used to multiplex and demultiplex the sequencing in one run. The amount of sampling from each cell is computed based on the results of previous iterations. This is called resource allocation and is similar to the one proposed by Haupt *et al*. (2011) and Wei and Hero (2012). The resulting read datasets, one per subgroup, are then co-assembled and compared using HyDA to decide pairwise subsumption of subgroups. The cells in those subgroups that are subsumed by other subgroups, even in previous iterations, are eliminated from further analysis. This procedure is continued until each remaining group contains only one cell. The resulting non-redundant single element groups capture all of the distinct genome sequences.

3 METHODS

Although distinct genomes are often identified as different species, there are numerous cases where distinct genomes are categorized as varied instances of the same species or even the same strain. Instead of identification of strains and species that are currently phenotypic notions, the goal of our approach is to find all distinct genomes in a sample. We define two genome sequences to be distinct if the ratio of their differences over the whole genome size is above a threshold. That threshold is input by the user and controls a trade-off between sensitivity and specificity.

Co-assembly and comparison of multiple input read datasets lie at the core of both approaches we take in this article. Although there are assembly tools for single-cell genomic data, such as SPAdes (Bankevich *et al*., 2012) and IDBA-UD (Peng *et al*., 2012), and also co-assembly tools for normal multicell genomic data such as Cortex (Iqbal *et al*., 2012), we use HyDA, which is the only tool to date that has both functionalities (Movahedi *et al*., 2012). However, the novel ideas proposed here can also be implemented using other assemblers. For the sake of completeness, HyDA algorithm is summarized in the following.

3.1 Co-assembly algorithm

3.1.1 Construction and condensation of the colored de Bruijn graph The colored de Bruijn graph, a variation of the standard de Bruijn graph, is a combinatorial structure that can be used to assemble a number of input read datasets, each represented by a color, superimposed on a single de Bruin graph (Iqbal *et al*., 2012). The output of such assembly methods is a number of assembled sequences (contigs) and the corresponding average multiplicity in each color. Our de Bruijn graph of the input reads is stored in a hashed collection of splay trees whose vertices are k-mers with an array of multiplicity counts (one entry per color), in- and out-edges and internal flags. A 1-in 1-out chain of k-mers is condensed into an equivalent long sequence, which is called a *unitig*. A maximal unitig, which cannot be extended further because of a branch in the graph, is a *contig*. Note that in HyDA, our condensation is solely based on the topology of the graph without any attention to the colored multiplicities. Ignoring multiplicities for condensation is purposefully done, and constitutes the feature that allows the assembler to deal with black out regions in single-cell MDA (Chitsaz *et al*., 2011).

Contigs with low coverage are often caused by sequencing error. The low-coverage contig removal process is iterated with an increased cutoff in each round. In each iteration, those contigs whose maximum coverage over all colors is less than the cutoff are eliminated, and the remaining graph is recondensed. This causes some contigs to merge into larger ones with recomputed average coverages. This process is similar to Velvet-SC's low-coverage contig removal, but instead of considering one average coverage per contig, HyDA considers the *maximum* of average coverages for all the colors of each contig (Chitsaz *et al*., 2011). In this case, only those contigs that have low coverage in all colors are considered erroneous and removed. Another possible approach is to eliminate those contigs for which the mean of average coverages for all colors is less than the cutoff. However, if we were to follow this approach, a contig that is well covered in one color but is poorly covered or absent in hundreds of colors would be lost, as the mean would dilute the effect of coverage in one color among hundreds of colors. This approach would not work for us here because our goal is to be able to preserve rare contigs.

3.1.2 Redundancy removal To remove redundant genomes, we define a relation that is reminiscent of subset relation on the set of contigs for each color. Note that our goal here is to remove redundant genomes, which are collections of contigs, rather than to remove individually redundant contigs. Let $A = \{a_1, a_2, \ldots, a_r\}$ be the set of remaining contigs after iterative error removal. Let $M_j(a_i)$ denote the average coverage of contig a_i in color j, for $1 \leq i \leq r$ and $1 \leq j \leq s$, where s is the number of colors. Pick $\epsilon \geq 0$, and let $A_j = \{a_i \in A \mid M_j(a_i) > \epsilon\} \subset A$ be

the set of contigs for each color j. The parameter ϵ determines the trade-off between specificity and sensitivity. We chose $\epsilon = 0$ in this study, but a non-zero ϵ might be needed if there are erroneous or contaminant k-mers in one color that also occur in the true genomic sequence of another color.

We define $D_\tau(A_i, A_j) \in \mathbb{R}$ on the set $\mathcal{F} = \{A_1, A_2, \ldots, A_s\}$ as:

$$D_\tau(A_i, A_j) = \tau - \frac{||A_i \backslash A_j||}{||A_i||}, \quad (1)$$

in which $A_i \backslash A_j = \{a \in A_i | a \notin A_j\}$, and $||\cdot||$ denotes the total assembly size. We define:

$$A_i \preceq_\tau A_j \text{ iff } 0 \leq D_\tau(A_i, A_j), \quad (2)$$

in which $\tau \geq 0$. Particularly, $\preceq_0$ becomes the subset relation and can be used to detect and remove redundant collections of contigs, i.e. those that are subsumed by a larger collection. However, in reality, the mathematical subset relation is not adequate as there are various types of noise, including sequencing errors, intrastrain variations such as single nucleotide polymorphisms and indels, contaminations added in the amplification and sequencing process and lack of coverage in some regions caused by the MDA. Hence, the definition of subset should be loosened by choosing a small but non-zero value for τ. Beside ϵ, the value of τ gives a trade-off between specificity and sensitivity of recognizing distinct genomes. If τ is small, the algorithm detects two equivalent genomes as distinct, and if τ is large, distinct genomes are declared equivalent. To see how τ is chosen, refer to Sections 3.2 and 4.2. The results are shown in Table 3. Finally, we compute a non-redundant set of assemblies $\mathcal{E} = \{A_{i_1}, A_{i_2}, \ldots, A_{i_t}\} \subseteq \mathcal{F}$, such that for every distinct pair $1 \leq a, b \leq t$, $A_{i_a} \not\preceq_\tau A_{i_b}$ and $A_{i_b} \not\preceq_\tau A_{i_a}$.

3.2 Divide and conquer strategy exploiting sparsity

Let n be the number of cells in the sample, and denote the cells by $S^i, i = 1, \ldots, n$. Our algorithm aims to assemble all of the distinct genomes and identify at least one cell per distinct genome. To reach this goal, our algorithm iteratively pools samples of amplicons from different cells, tags each pool with a unique barcode, mixes the barcoded pools and has the result sequenced. The objective is to minimize the total number of bases required to be sequenced as well as the number of different barcodes needed.

In the first iteration, we divide the n cells $S^1, \ldots, S^n$ into two sets

$$I_{1,1} = \{S^1, \ldots, S^{\lfloor n/2 \rfloor}\},$$

$$I_{1,2} = \{S^{\lfloor n/2 \rfloor + 1}, \ldots, S^n\}.$$

Our algorithm samples equal amount of amplicons from each cell in $I_{1,1}$ and $I_{1,2}$. The amplicons in each set are pooled and tagged by two distinct barcodes. The barcoded amplicons are sequenced to reach a desired number of base pairs. This number is an input parameter of our algorithm. We define the total number of base pairs sequenced from $I_{1,i}$ to be $b_{1,i}$, for $i = 1, 2$. The two read datasets are co-assembled by HyDA using two colors. The result is two sets of contigs for each color, $A_{1,1}$ and $A_{1,2}$. We calculate $D_{\tau_1}(A_{1,1}, A_{1,2})$ and $D_{\tau_1}(A_{1,2}, A_{1,1})$ as defined in (1), in which $\tau_1 = \tau / \max_j |I_{1,j}|$, τ is an input parameter and $|\cdot|$ is the set cardinality. We choose τ to be the maximum allowable difference between the assembly of two single cells from the same strain. Based on these values, we decide whether the relations $A_{1,1} \preceq_{\tau_1} A_{1,2}$ and $A_{1,2} \preceq_{\tau_1} A_{1,1}$ hold. If $A_{1,1}$ is a subset of $A_{1,2}$, then all of the distinct genomes in $I_{1,1}$ are present in $I_{1,2}$; therefore, the cells in $I_{1,1}$ do not need further sampling. This applies to $I_{1,2}$ too, if $A_{1,2}$ is a subset of $A_{1,1}$. If both relations hold, one of $I_{1,1}$ or $I_{1,2}$ is eliminated arbitrarily from further analysis. Each remaining set $I_{1,\cdot}$ is divided into two subsets for analysis in the second iteration. Figure 1 depicts an example of 10 cells with 3 distinct genomes shown in different colors.

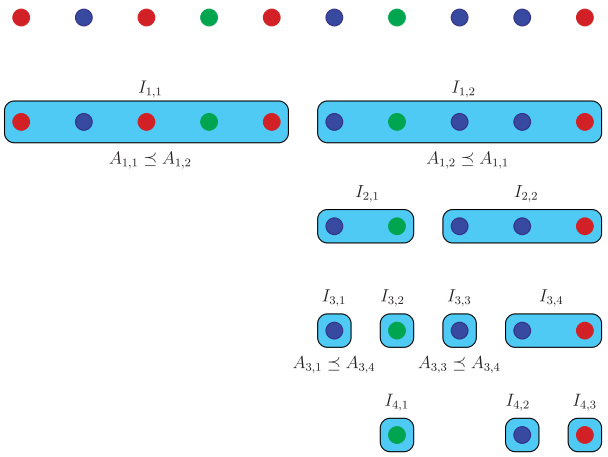

Fig. 1. The divide and conquer algorithm for an example with 10 cells and 3 distinct genomes shown in different colors. Each row corresponds to one sequencing round. The number of barcodes in each round is the number of blue boxes in the corresponding row

The same splitting process occurs in the subsequent iterations. Assume $I_{i,1}, \ldots, I_{i,m_i}$ are the remaining sets in iteration i. Each set $I_{i,j}$ is sampled to produce $b_{i,j}$ base pairs, barcoded uniquely, pooled and sequenced. All of the new sequence datasets and those obtained in all previous iterations are co-assembled. In the co-assembly, the previous datasets help to improve the assembly of the new ones. The resulting contig set of $I_{i,j}$ is denoted by $A_{i,j}$. For all $j, k = 1 \ldots m_i$, the relations $A_{i,j} \preceq_{\tau_i} A_{i,k}$ are evaluated, where τ_i is a threshold whose calculation will be explained below. The cells in those $I_{i,j}$ whose assemblies are subsumed will be removed from further analysis. All the remaining ones are partitioned into two disjoint subsets. Denote the new subsets by $I_{i+1,1}, \ldots, I_{i+1,m_{i+1}}$. Note that in iteration i, m_i unique barcodes are needed. Therefore,

$$m = \max_i m_i \quad (3)$$

is the maximum number of barcodes required for the entire algorithm.

Parameters $b_{i,j}$ play a key role in the algorithm. We propose an adaptive calculation of $b_{i,j}$ to minimize, without losing accuracy, the total base pairs sequenced:

$$b = \sum_{i,j} b_{i,j}. \quad (4)$$

Assume $I_{i,j}$ is a set that is created by dividing the set $I_{i-1,k}$ in iteration $i-1$ into two. We are motivated to choose the total number of sequenced base pairs from cells in $I_{i,j}$ to be proportional to the total assembly size $||A_{i-1,k}||$, i.e.

$$b_{i,j} = c \times ||A_{i-1,k}||, \quad (5)$$

where c is an input parameter indicating the estimated average coverage of each iteration. We say $A_{i,j}$ are *accurate enough* if the partial order relation $\preceq_{\tau_i}$ can be assessed accurately. If c is large and the assembly of iteration $i-1$ is accurate enough, then in iteration i, adequate base pairs are sequenced to allow an accurate enough assembly of set $I_{i,j}$.

Another factor that affects accuracy of the assessment of these relations is the choice of τ_i. The threshold τ_i in the i^{th} iteration is used to detect cells with similar genomes despite small differences in their assemblies. We propose to use the following threshold:

$$\tau_i = \frac{\tau}{\max_{1 \leq j \leq m_i} |I_{i,j}|}. \quad (6)$$

Recall that τ is the maximum allowable difference between the assembly of two single cells with similar genome sequences. To account for the

worst possible case, it is assumed that there are $|I_{i,j}|$ distinct genomes in each group $I_{i,j}$. Therefore, $\max_{1 \le j \le m_i} |I_{i,j}|$ captures the maximum number of distinct genomes in $I_{i,j}$ from any $I_{i,k}$. With these assumptions, τ_i is a conservative threshold. This threshold will guarantee that two distinct genomes are detected, but it has the possibility of detecting similar genomes as distinct. In the last iteration of the algorithm, when every group consists of one cell, the threshold is τ. Note that the number of iterations, which is the number of sequencing rounds, is always $\lceil \log_2 n \rceil$.

3.3 Implementation

We implemented our algorithm in a tool called `Squeezambler 1.0` in C++. Our tool and datasets are available at http://compbio.cs.wayne.edu/software/squeezambler/.

4 RESULTS

Because we did not have access to real data, we tested our algorithm using simulated data. We used our tool `MDAsim 1.0` (Taghavi and Draghici, 2012) to simulate 100 MDA processes (one process per cell) from 9 distinct genomes. The output of `MDAsim 1.0` was fed into an Illumina read generator, `ART` (Huang *et al.*, 2012), to generate Illumina reads, with realistic errors, from the simulated amplicons. The set of generated reads for each cell was treated like a microfluidic droplet from which samples without replacement are extracted in each iteration of `Squeezambler 1.0`. We assume that MDA products are contamination-free, which require a contaminant-free automated microfluidic cell sorting, amplification and sampling device.

4.1 Datasets

A total of 115 MDAs were simulated from nine distinct genomes chosen from the list of species found in a gut metagenomics study (Qin *et al.*, 2010) that have a complete or draft reference genome. Recall that we are simulating MDA from a reference genome; therefore, we needed a reference genome for the chosen species. Table 1 summarizes the identification number in the National Center for Biotechnology Information database (NCBI ID), name, size, reference status (complete or draft) and the number of cells we have simulated. The number of cells is approximately proportional to the abundance mean of the corresponding species in Qin *et al.* (2010). We ran `MDAsim 1.0` with a diverse set of parameters, one for each cell, to capture the diverse nature of MDA coverage bias. `ART`, an Illumina read generator, was then deployed to generate 100 bp Illumina reads from the simulated amplicons. The amplification gain of `MDAsim 1.0` was 300× and that of `ART` was 8× from which one-eighth were selected randomly to obtain a total gain of 300×. We assembled the dataset of each cell individually with `HyDA 1.0`, and the resulting assemblies have between 0.1 and 4.2% missing reference bases measured by `Gage` (Salzberg *et al.*, 2012), which is similar to the real world situation with a successful MDA reaction (Chitsaz *et al.*, 2011). We made an error profile that matches the error statistics of Illumina HiSeq 2000 for `ART`. Using our profile, `ART` injects on average 1% error into the reads, because of which we need to eliminate erroneous contigs in the assembler. `HyDA 1.0`, and also its predecessor `Velvet-SC`, has an iterative algorithm to remove low-coverage contigs, which is explained in Section 3.1.1. In each iteration, `Squeezambler 1.0` provides `HyDA 1.0` with a coverage cutoff as a percentage of the mean coverage of each color. That percentage is constant in all iterations.

We designed three collections of cells, the statistics of which are summarized in Table 2. In the first collection, there are 62 cells with 6 distinct genomes. In this collection, we put 22 different MDA results from NC_004663.1 and 22 from FP929051.1 to

Table 2. Our simulation setups: (i) 62 cells; 6 distinct genomes, (ii) 97 cells; 5 distinct genomes and (iii) 112 cells; 7 distinct genomes

NCBI ID	Abundance (%)					
	62 cells; 6 distinct genomes		97 cells; 5 distinct genomes		112 cells; 7 distinct genomes	
NC_004663.1	22	36%	23	24%	23	21%
NC_009614.1	7	11%	0	0%	7	6%
NC_009615.1	8	13%	0	0%	8	7%
NC_008532.1	2	3%	0	0%	0	0%
NC_016776.1	1	1%	0	0%	0	0%
FP929046.1	0	0%	12	12%	12	11%
FP929051.1	22	36%	35	36%	35	31%
FP929053.1	0	0%	12	12%	12	11%
FP929055.1	0	0%	15	16%	15	13%

Table 1. The nine chosen distinct genomes (species) for our simulation

NCBI ID	Name	Reference status	Size (Mbp)	Number of cells
NC_004663.1	*Bacteroides thetaiotaomicron* VPI-5482 chromosome	Complete	6.29	23
NC_009614.1	*Bacteroides vulgatus* ATCC 8482 chromosome	Complete	5.16	7
NC_009615.1	*Parabacteroides distasonis* ATCC 8503 chromosome	Complete	4.81	8
NC_008532.1	*Streptococcus thermophilus* LMD-9	Complete	1.86	2
NC_016776.1	*Bacteroides fragilis* 638R	Complete	5.37	1
FP929046.1	*Faecalibacterium prausnitzii* SL3/3	Draft	3.21	12
FP929051.1	*Ruminococcus bromii* L2-63	Draft	2.25	35
FP929053.1	*Ruminococcus sp.* SR1/5	Draft	3.55	12
FP929055.1	*Ruminococcus torques* L2-14	Draft	3.34	15

play the role of highly abundant genomes in a sample as well as 1 from NC_016776.1 and 2 from NC_008532.1 to represent low-abundance genomes in the same sample. In the first collection, the number of distinct genomes is approximately one-tenth of the number of cells, but with a wide range of abundances. The second collection is the sparsest collection among the three, where the number of distinct genomes is approximately one-twentieth of the number of cells. The third collection is the most diverse of the three, where the number of distinct genomes is approximately one-sixteenth of the number of cells.

4.2 Simulation results

We ran `Squeezambler 1.0` for the three collections described above, the results of which are summarized in Table 3. Most of the `Squeezambler 1.0` parameters were the same for all three collections. The assembly inclusion threshold constant per cell was chosen $\tau = 0.2$, which means at most 20% of the assembly can vary among multiple instances of the same genome. Taking into account the genomic sequence loss caused by the MDA, sampling of the amplicons and sequencing errors, 20% is a reasonable choice. This is not an optimized value and is chosen based on the authors' intuition. We chose τ conservatively in this work so that distinct genomes are not lost but some equivalent genomes are detected as distinct. Finding the optimal value for τ needs a thorough study, which is beyond the scope of this article.

The k-mer size for `HyDA 1.0` was chosen to be $k = 55$, which is a common choice for the chosen Illumina error profile (Chitsaz *et al.*, 2011). The coverage cutoff, as a percentage of the coverage mean, was chosen to be 100%, and the minimum contig length was 100 bp for `HyDA 1.0`. The coverage mean is estimated based on the assembly size in the first iteration, which is often larger than the actual genome size because of a myriad of erroneous low coverage k-mers. This causes the initially estimated coverage mean to be a small fraction of the final coverage mean after error removal.

`Squeezambler 1.0` has an option to choose the number of initial groups in the first iteration, g. If g is chosen to be equal to the number of cells, then `Squeezambler 1.0` simulates the single-cell co-assembly of all the cells. If $g = 2$, then `Squeezambler 1.0` simulates the divide and conquer algorithm described in Section 3.2. Although experimenting with different g values may improve the results, we do not have data for it.

Before any sequencing is done, the algorithm has no idea about the genome sizes, various distinct genomes and abundance of each genome. Therefore, an unbiased algorithm has to sequence from each cell exactly the same amount right in the first iteration. That amount in our algorithm, denoted by $b_{1,j}/|I_{1,j}|$, is an input parameter to `Squeezambler 1.0`, which was chosen to be between 1 Mbp and 63 Mbp as reported in the third column of Table 3. In our setup, the size of the nine distinct genomes varies between 1.8 and 6.3 Mbp; see Table 1. Therefore, 1 Mbp sequencing provides between 1/6× and 1/2× coverage, and 63 Mbp sequencing provides between 10× and 35× coverage.

The input parameter c, which controls the amount of sequencing in subsequent rounds, was chosen to be $c = 10$, which means 10× expected coverage from each genome in each collection. We observed that in practice 10× coverage of each distinct genome provides sufficient information for reliable evaluation of $\preceq$. This is consistent with the Lander and Waterman (1988) analysis, in which the statistics of gaps and contigs in terms of coverage is characterized. Based on that analysis, 10× coverage yields the entire genome without gap with high probability.

Our divide and conquer algorithm exhibits significant improvement in maximum barcodes, and in most cases the total number of base pairs sequenced, over the single-cell co-assembly method. For the 97 cells, 5 distinct genomes collection, our divide and conquer algorithm requires only 2.9 Gbp sequencing and 10 barcodes in comparison with 3.0 Gbp sequencing and 97 barcodes consumed by the single-cell co-assembly method. Similarly for the 62 cells, 6 distinct genomes collection, our divide and conquer algorithm requires only 3.0 Gbp sequencing and 10 barcodes in comparison with 3.9 Gbp sequencing and 62 barcodes required by the single-cell co-assembly method.

Table 3. `Squeezambler 1.0` results for the three setups summarized in Table 2

Setup	Method	Sequencing per cell in the first iteration[a] (Mbp)	Total sequencing[b] (Gbp)	Max barcodes[c]	Number of predicted distinct genomes	Iterations
62 cells; 6 distinct genomes	Single-cell co-assembly	63	3.9	62	6	1
	Divide and conquer	7	**3.0**	**10**	8	6
97 cells; 5 distinct genomes	Single-cell co-assembly	63	6.1	97	5	1
	Single-cell co-assembly	31	3.0	97	11	1
	Divide and conquer	1	3.2	17	5	7
	Divide and conquer	7	**2.9**	**10**	6	7
112 cells; 7 distinct genomes	Single-cell co-assembly	63	7.1	112	7	1
	Single-cell co-assembly	31	**3.5**	112	14	1
	Divide and conquer	1	7.1	**33**	11	7

Note: We report the results for two different values of initial sequencing coverage per cell for some methods. Bold numbers indicate the best results for the corresponding parameters for each setup.

[a] $b_{1,j}/|I_{1,j}|$.

[b] b in (4).

[c] m in (3).

Even though for the 112 cells, 7 distinct genomes collection, our divide and conquer algorithm outperforms single-cell co-assembly in terms of the number of barcodes, by 33 versus 112, it requires 7.1 Gbp sequencing, which is more than that used by single-cell co-assembly (3.5 Gbp).

In all of our experiments, all distinct genomes were correctly detected. Therefore, our results exhibit ultimate sensitivity. However, in some experiments, multiple cells with similar genomes were identified as distinct, which is not an issue for our problem, because, based on the results of `Squeezambler 1.0`, those cells that are identified as with distinct genomes can be deeply sequenced and assembled afterward. For the 62 cells, 6 distinct genomes collection, the number of detected distinct genomes was between 6 and 8. For the 97 cells, 5 distinct genomes collection, that number was between 5 and 11, and for the 112 cells, 7 distinct genomes collection, that was between 7 and 14. This specificity is reported in the sixth column of Table 3. Note that the number of sequencing rounds (iterations) for single-cell co-assembly is always 1, and for divide and conquer with $g = 2$, it is $\lceil \log_2 n \rceil$.

Owing to the computational intensity of `MDAsim 1.0`, `HyDA 1.0` and `Squeezambler 1.0`, we report our results for only small examples to provide a proof of concept. We also chose our parameters conservatively, and without optimization, so that we do not compromise accuracy. Moreover, our examples are in the order of 100 cells and 6 distinct genomes, whereas real world samples are much sparser, as the number of cells may be in the order of billions and the number of distinct genomes at most in the order of thousands. Therefore, we expect the reduction in the total sequencing and maximum barcodes to be higher for real world applications than what we report in this article.

5 CONCLUSION

We presented an adaptive divide and conquer algorithm for distilled sequencing and *de novo* assembly of distinct genomes in a bacterial community, e.g. human gut microbiome. Samples derived from such communities are often sparse in the sense that the number of distinct genomes is much less than the number of cells. Our algorithm exploits sparsity to decrease the amount of sequencing and the number of multiplexing barcodes needed for single-cell sequencing and *de novo* assembly.

We implemented our algorithm in a tool which we call `Squeezambler` and performed simulation experiments to demonstrate its power. Our results show that (i) the number of required barcodes with our divide and conquer algorithm is less than that required by the naïve approach, and that (ii) the amount of sequencing needed remains the same or decreases. Owing to the computational intensity of the problem, only small examples with low sparsity were studied in this work. Real-world samples are much sparser ($\sim$1000 species in $\sim 10^{14}$ cells) than the examples here ($\sim$5 species in $\sim$100 cells). Also, the parameters used to run our tool were chosen conservatively and without optimization. Therefore, we expect the improvement of our algorithm to be higher than what we reported in this article in real-world situations. `Squeezambler 1.0` identifies all distinct genomes in the sample, which are candidates for different strains/species. Those cells that are identified as having distinct genomes need to be subsequently deeply sequenced and assembled to obtain a more detailed assembly.

Funding: NIH RO1 DK089167, STTR R42GM087013 and NSF DBI-0965741 (to S.D.).

Conflict of Interest: none declared.

REFERENCES

Bankevich,A. *et al.* (2012) SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J. Comput. Biol.*, **19**, 455–477.

Candès,E. and Tao,T. (2005) Decoding by linear programming. *IEEE Trans. Inf. Theory*, **51**, 4203–4215.

Candès,E. and Tao,T. (2006) Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inf. Theory*, **52**, 5406–5425.

Chitsaz,H. *et al.* (2011) Efficient *de novo* assembly of single-cell bacterial genomes from short-read data sets. *Nat. Biotechnol.*, **29**, 915–921.

Dean,F.B. *et al.* (2001) Rapid amplification of plasmid and phage DNA using Phi 29 DNA polymerase and multiply-primed rolling circle amplification. *Genome Res.*, **11**, 1095–1099.

Dean,F.B. *et al.* (2002) Comprehensive human genome amplification using multiple displacement amplification. *Proc. Natl Acad. Sci. USA*, **99**, 5261–5266.

Donoho,D. (2006) Compressed sensing. *IEEE Trans. Inf. Theory*, **52**, 1289–1306.

Erlich,Y. *et al.* (2010) Compressed genotyping. *IEEE Trans. Inf. Theory*, **56**, 706–723.

Haupt,J. *et al.* (2011) Distilled sensing: adaptive sampling for sparse detection and estimation. *IEEE Trans. Inf. Theory*, **57**, 6222–6235.

Hongoh,Y. *et al.* (2008) Complete genome of the uncultured Termite Group 1 bacteria in a single host protist cell. *Proc. Natl Acad. Sci. USA*, **105**, 5555–5560.

Hosono,S. *et al.* (2003) Unbiased whole-genome amplification directly from clinical samples. *Genome Res.*, **13**, 954–964.

Huang,W. *et al.* (2012) ART: a next-generation sequencing read simulator. *Bioinformatics*, **28**, 593–594.

Iqbal,Z. *et al.* (2012) *De novo* assembly and genotyping of variants using colored de bruijn graphs. *Nat. Genet.*, **44**, 226–232.

Ishoey,T. *et al.* (2008) Genomic sequencing of single microbial cells from environmental samples. *Curr. Opin. Microbiol.*, **11**, 198–204.

Kvist,T. *et al.* (2007) Specific single-cell isolation and genomic amplification of uncultured microorganisms. *Appl. Microbiol. Biotechnol.*, **74**, 926–935.

Lander,E.S. and Waterman,M.S. (1988) Genomic mapping by fingerprinting random clones: a mathematical analysis. *Genomics*, **2**, 231–239.

Lasken,R.S. (2007) Single-cell genomic sequencing using Multiple Displacement Amplification. *Curr. Opin. Microbiol.*, **10**, 510–516.

Lasken,R.S. and Stockwell,T.B. (2007) Mechanism of chimera formation during the Multiple Displacement Amplification reaction. *BMC Biotechnol.*, **7**, 19.

Marcy,Y. *et al.* (2007) Dissecting biological "dark matter" with single-cell genetic analysis of rare and uncultivated TM7 microbes from the human mouth. *Proc. Natl Acad. Sci. USA*, **104**, 11889–11894.

Methe,B.A. *et al.* (2012) A framework for human microbiome research. *Nature*, **486**, 215–221.

Movahedi,N.S. *et al.* (2012) *De novo* co-assembly of bacterial genomes from multiple single cells. In: *IEEE Conference on Bioinformatics and Biomedicine*. Philadelphia, PA, USA, pp. 561–565.

Mussmann,M. *et al.* (2007) Insights into the genome of large sulfur bacteria revealed by analysis of single filaments. *PLoS Biol.*, **5**, e230.

Peng,Y. *et al.* (2012) IDBA-UD: a *de novo* assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics*, **28**, 1420–1428.

Podar,M. *et al.* (2007) Targeted access to the genomes of low-abundance organisms in complex microbial communities. *Appl. Environ. Microbiol.*, **73**, 3205–3214.

Qin,J. *et al.* (2010) A human gut microbial gene catalogue established by metagenomic sequencing. *Nature*, **464**, 59–65.

Raghunathan,A. *et al.* (2005) Genomic DNA amplification from a single bacterium. *Appl. Environ. Microbiol.*, **71**, 3342–3347.

Rodrigue,S. *et al.* (2009) Whole genome amplification and *de novo* assembly of single bacterial cells. *PLoS One*, **4**, e6864.

Salzberg,S.L. *et al.* (2012) GAGE: a critical evaluation of genome assemblies and assembly algorithms. *Genome Res.*, **22**, 557–567.

Stobbe,P. and Krause,A. (2012) Learning fourier sparse set functions. *J. Mach. Learn. Res.*, **22**, 1125–1133.

Taghavi,Z. and Draghici,S. (2012) Mdasim: a multiple displacement amplification simulator. In: *IEEE Conference on Bioinformatics and Biomedicine*. Philadelphia, PA, USA, pp. 575–578.

Treangen,T.J. *et al.* (2013) MetAMOS: a modular and open source metagenomic assembly and analysis pipeline. *Genome Biol.*, **14**, R2.

Wei,D. and Hero,A. (2012) Multistage adaptive estimation of sparse signals. In: *IEEE Statistical Signal Processing Workshop (SSP)*. Ann Arbor, MI, USA, pp. 153–156.

Woyke,T. *et al.* (2009) Assembling the marine metagenome, one cell at a time. *PLoS One*, **4**, e5299.

Zhang,K. *et al.* (2006) Sequencing genomes from single cells by polymerase cloning. *Nat. Biotechnol.*, **24**, 680–686.