F1000Research Evaluating Metagenome Assembly on a Complex Community

Sherine Awad[1], **Luiz Irber**[2], **and C. Titus Brown**[3]

[1,2,3]**Department of Population Health and Reproduction, University of California, Davis, California**

Abstract

Metagenome assembly is a challenging problem due to the biodiversity of the microorganisms. Most assemblers are designed for whole genome assembly and not capable of dealing with metagenomic samples. However, in order to decide which assembler works best for metagenome, we need to evaluate metagenome assembly generated by each assembler.

In this paper, we used three assemblers ; IDBA-UD, SPAdes, and Megahit to assemble metagenome mock community data and evaluate the assembly process in terms of resources utilization, assembly quality, genome fraction covered, duplication ratio, misassemblies and partial alignments.

The results show only small differences in content recovery between assemblers. However, Megahit is much faster and produces shorter contig lengths than IDBA-UD and SPAdes.

Introduction

Metagenomics refers to sequencing of DNA from a mixture of organisms, often from an environmental or uncultured sample. Unlike whole genome sequencing, metagenomics targets a mixture of genomes, which introduces metagenome-specific challenges in analysis. Most approaches to analyzing metagenomic data rely on mapping or comparing sequencing reads to reference sequence collections. However, reference databases contain only a small subset of microbial diversity (cite: geba), and the much of the remaining diversity is evolutionarily distant and search techniques may not access it.

As sequencing capacity increases and sequence data is generated from many more environmental samples, metagenomics is increasingly using de novo assembly techniques to generate new reference genomes and metagenomes. There are a number of metagenome assemblers that are widely used. However, evaluating the results of these assemblers is challenging due to the general lack of good quality reference metagenomes. Below, we evaluate three commonly assemblers - SPAdes, IDBA, and MEGAHIT - on a mock community containing 64 species of microbes with known genomes.

Moya et al. in [**?**] evaluated metagenome assembly using simulated two 454 viral metagenome and six assemblers. The assemblies were evaluated based on several metrics including N50, percentages of reads assembled, accuracy when compared to the reference genome. In addition to, chimeras per contigs and the effect of assembly on taxonomic and functional annotations.

Mavromatis et al. in [**?**] provided a benchmark study to evaluate the fidelity of metagenome process methods. The study used simulated metagenomic data sets constructed at different complexity levels. The datasets were assembled using Phrap v3.57, Arachne v.2 [**?**] and JAZZ. [**?**]

The study evaluates assembly, gene prediction, and binning methods. However, the study did not evaluate the assembly quality against a reference genome.

Rangwala et al. in [**?**] presented an evaluation study of metagenome assembly. The study used a de Bruijn graph based assembler ABYSS [**?**] to assemble simulated metagnome reads of 36 bp. The data set is classified at different complexity levels. The study compares the quality of the assembly of the data sets in terms of quality measures of contigs length, assembly accuracy. The study also took into consideration the effect of kmer size and the degree of chimericity. However, the study evaluated the assembly based on one assembler, and did not evaluate assembly against several assemblers. Also, both previous studies used simulated data, which may lack confounders of assembly such as sequencing artifacts and GC bias.

Lindgreeb et al in [**?**] presented a benchmark study for metagenome analysis tools. The authors compared between several metagenome classification tools in terms of run time, ease of use, information provided, reads and shuffled reads mapped, non existing phyla, divergence of real distribution, and correlation with known community composition. However, the paper did not consider metagenome assembly tools in the study.

In this paper, we evaluate metagenome assembly on the data set from Shakya et al. (2013) using three assemblers; IDBA-UD [**?**], SPAdes [**?**], and MEGAHIT [**?**].

SPAdes [**?**] is an assembler for both single-cell and standard (multicell) assembly. (More description here @CTB.)

IDBA-UD [**?**] is a de Bruijn graph approach for assembling reads from single cell sequencing or metagenomic sequencing technologies with uneven sequencing depths. IDBA-UD uses multiple depth-relative thresholds to remove erroneous k-mers in both low-depth and high-depth regions. It also uses paired-end information to solve the branch problem of low-depth short repeat regions. It also applies an error correction step to correct reads of high-depth regions that can be aligned to high confident contigs.

MEGAHIT [**?**] is a newer approach that constructs a succinct de Bruijn graph using multiple k-mer sizes, and uses a novel “mercy k-mer” approach that preserves low-abundance regions of reads. It also can use GPUs to accelerate the graph construction.

We evaluate the performance of the three assemblers using real mock community data. The performance of each assembler is compared in terms of resources utilization, covered genome fraction, duplication ratio, misassemblies, and contig length. This helps decide which assembler to use when we lack a reference.

Table 1. Running Time and Memory Utilization

(1) IDBA-UD	
Running Time	17:12:43
Memory Utilization (GB)	149.12
(2) SPAdes	
Running Time	42:14:06
Memory Utilization (GB)	391.45
(3) MEGAHIT	
Running Time	56:04.43
Memory Utilization (GB)	34.40

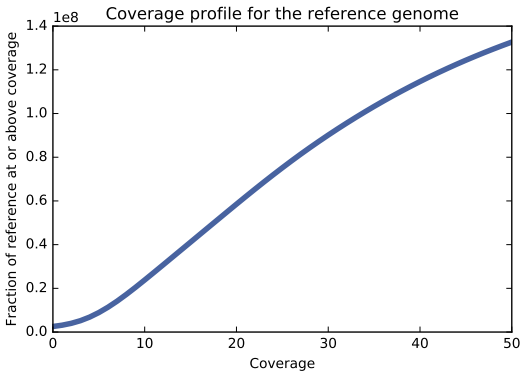

Figure 1. Cumulative coverage profile for the reference metagenome, based on read mapping.

Table 2. Reference Genome Coverage and Duplication Ratio

(1) Best hit Approach		
(1) IDBA-UD		
99.0	**Genome Coverage**	56.89 %
	Duplication Ratio	0.38 %
95.0	**Genome Coverage**	58.00%
	Duplication Ratio	0.59 %
(2) SPAdes		
99.0	**Genome Coverage**	63.79 %
	Duplication Ratio	0.15 %
95.0	**Genome Coverage**	64.68 %
	Duplication Ratio	0.26%
(3) MEGAHIT		
99.0	**Genome Coverage**	68.47 %
	Duplication Ratio	0.37 %
95.0	**Genome Coverage**	68.96 %
	Duplication Ratio	0.45%
(2) Ambiguous Approach		
(1) IDBA-UD		
99.0	**Genome Coverage**	89.79 %
	Duplication Ratio	0.94%
95.0	**Genome Coverage**	95.46 %
	Duplication Ratio	1.90 %
(2) SPAdes		
99.0	**Genome Coverage**	89.42 %
	Duplication Ratio	1.00 %
95.0	**Genome Coverage**	95.12 %
	Duplication Ratio	1.98 %
(3) MEGAHIT		
99.0	**Genome Coverage**	91.16 %
	Duplication Ratio	0.55 %
95.0	**Genome Coverage**	94.22 %
	Duplication Ratio	1.48 %
(3) No Misassemblies Approach		
(1) IDBA-UD		
99.0	**Genome Coverage**	34.60 %
	Duplication Ratio	0.37%
95.0	**Genome Coverage**	35.22 %
	Duplication Ratio	0.56 %
(2) SPAdes		
99.0	**Genome Coverage**	35.92%
	Duplication Ratio	0.16 %
95.0	**Genome Coverage**	36.42 %
	Duplication Ratio	0.21 %
(3) MEGAHIT		
99.0	**Genome Coverage**	45.81 %
	Duplication Ratio	0.39%
95.0	**Genome Coverage**	46.03 %
	Duplication Ratio	0.48 %

Table 3. Contigs Analysis

(1) Best hit Approach	
(1) IDBA-UD	
No. of Contigs	19,988
Totally Aligned Contigs %	72.96% (97,138,779)
Partial Aligned Contigs %	10.97% (20,261,669)
Unaligned Contigs %	16.07% (61,421,243)
(2) SPAdes	
No. of Contigs	15,254
Totally Aligned Contigs%	76.52% (109,342,809)
Partial Aligned Contigs%	12.08% (22,008,234)
Unaligned Contigs%	11.40% (34,176,209)
(3) MEGAHIT	
No. of Contigs	27,657
Totally Aligned Contigs %	83.07% (128,987,917)
Partial Aligned Contigs%	4.67% (12,325,804)
Unaligned Contigs%	12.26% (50,093,476)
(2) Ambiguous Approach	
(1) IDBA-UD	
No. of Contigs	19,988
Totally Aligned Contigs%	80.59% (161,075,933)
Partial Aligned Contigs%	8.54 % (22,638,415)
Unaligned Contigs %	10.87% (13,378,572)
(2) SPAdes	
No. of Contigs	15,254
Totally Aligned Contigs	81.03 % (154,920,366)
Partial Aligned Contigs%	9.34% (28,028,529)
Unaligned Contigs%	9.62% (12,931,934)
(3) MEGAHIT	
No. of Contigs	27,657%
Totally Aligned Contigs%	87.59 % (169,789,173)
Partial Aligned Contigs%	4.36% (15,658,616)
Unaligned Contigs%	8.04% (12,777,886)
(3) No Misassemblies Approach	
(1) IDBA-UD	
No. of Contigs	19,988
Totally Aligned Contigs%	57.41% (61,874,288)
Partial Aligned Contigs%	6.81%(9,530,232)
Unaligned Contigs %	35.78% (122,899,406)
(2) SPAdes	
No. of Contigs	15,254
Totally Aligned Contigs	62.93% (64,656,150)
Partial Aligned Contigs%	6.84% (9,323,121)
Unaligned Contigs%	30.23% (114,292,858)
(3) MEGAHIT	
No. of Contigs	27,657
Totally Aligned Contigs%	67.62% (87,488,973)
Partial Aligned Contigs%	2.73% (7,069,258)
Unaligned Contigs%	29.65% (102,672,613)

Datasets

We used a diverse mock community data set constructed by pooling DNA from 64 species of bacteria and archaea and sequencing them with Illumina HiSeq. The raw data set consisted of 109,629,496 reads from Illumina HiSeq 101 bp paired-end sequencing (2x101) with an

Table 4. Genomes with the most common uncovered bases between the three assemblies.

Genome	Uncovered bases (% of total)
Shewanella baltica OS223	2.3 Mbp (18.25%)
Fusobacterium nucleatum	2.2 Mbp (16.95%)
Desulfovibrio vulgaris DP4	1.6 Mbp (12.85%)
Enterococcus faecalis V583	1.0 Mbp (7.83%)
Thermus thermophilus HB27	0.7 Mbp (5.73%)

Table 5. Misassembled Contigs using Identity 99%

Assembler	No. of Contigs	No.of Bases
IDBA-UD	4,980 (24.91%)	112,309,828
SPAdes	3,143 (20.60%)	108,969,624
Megahit	5,977 (21.61%)	90,889,558

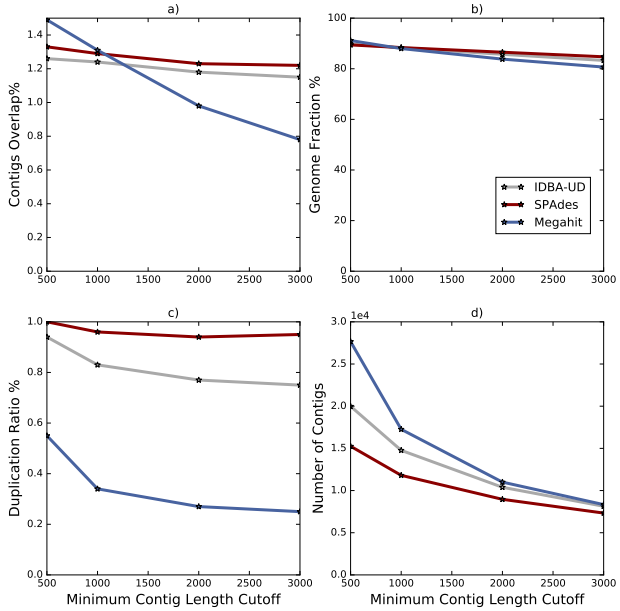

Figure 2. Genome fraction, duplication ratio, Contigs overlap ratio, and number of contigs using different minimum contig length, identity 99%, and Ambiguous Approach

untrimmed total length of 11.07 Gbp and an estimated fragment size of 380 bp [**?**].

The original reads are available through the NCBI Sequence Read Archive at Accession SRX200676. We received the 64 reference genomes from the original authors. They consist of 205.6 Mbp of assembled genomes in 64 contigs, and are available for download at https://dx.doi.org/10.6084/m9.figshare.1506873.v2

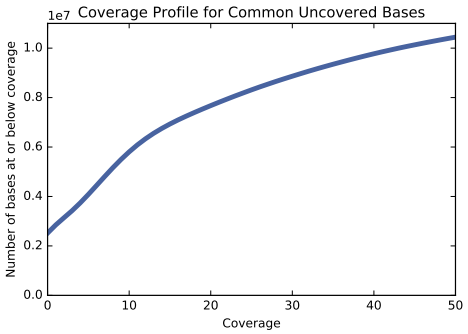

Figure 3. Cumulative read coverage for bases in the reference metagenome missing from all three assemblies.

Methods

The analysis code and run scripts for this paper are available at: https://github.com/dib-lab/2015-metagenome-assembly/.

Quality Filtering

We removed adapters with Trimmomatic v0.30 in paired-end mode with the Truseq adapters [**?**], using light quality score trimming as recommended in @cite MacManes 2014.

Reference Coverage Profile

To evaluate how much of the reference metagenome was contained in the read data, we used `bwa aln` to map reads to the reference genome. We then calculated how many reference bases were covered by how many mapped reads (custom script `coverage-profile.py`).

Assemblers

We assembled the quality-filtered reads using three different assemblers: IDBA-UD [**?**], SPAdes [**?**], and MEGAHIT [**?**]. For IDBA-UD v1.1.1 [**?**], we used `-pre_correction` to perform pre-correction before assembly and -r for the pe files.

For SPAdes v3.9.0 [**?**], we used `-meta -pe1-12 -pe1-s` where `-meta` is recommended when working with metagenomic data sets, `-pe1-12` specifies the interlaced reads for the first paired-end library, and `-pe1-s` provides the orphan reads remaining from quality trimming.

For MEGAHIT [**?**], we used -l 101 `-m 3e9 -cpu-only` where `-l` is for maximum read length, `-m` is for max memory in bytes to be used in constructing the graph, and `-cpu-only` to use only the CPU and not the GPU. We also used `-presets meta-large` for large and complex metagenomes, and `-12` and `-r` are parameters that specify the interleaved-paired-end and single-end files respectively.

All three assemblies were executed on the same high-memory buy-in node on the Michigan State University High Performance Compute Cluster, and we recorded

RAM and CPU time of each assembly job using the `qstat` utility at the end of each run.
Unless otherwise mentioned, we eliminated all contigs less than 500 bp from each assembly prior to further analysis.

Assembly analysis using Nucmer

We used the Nucmer tool from MUMmer3.23 [**?**] to align assemblies to the reference genome with options `--coords -p`. Then we parsed the generated ".coords" file using a custom script `analyze_assembly.py` to calculate several analysis metrics at two alignment identities, 95% and 99%.

Mapping

We aligned all quality-filtered reads to the reference metagenome with bwa aln (v0.7.7.r441) [**?**]. We aligned paired-end and orphaned reads separately using bwa aln samse. We then used samtools (v0.1.19) [**?**] to convert SAM files to BAM files for both paired-end and orphaned reads. To count the unaligned reads, we included only those records with the "4" flag in the SAM files [**?**].
To extract the reads that contribute to unaligned contigs, we mapped the quality filtered reads to the unaligned contigs using bwa aln (v0.7.7.r441) [**?**]. Then we used samtools to retrieve the reads that mapped to the unaligned contigs.

k-mer Presence

In order to examine k-mer presence for a k-mer size of 20, we build a k-mer counting table from the given quality filtered reads using `load-into-counting.py` from khmer (@cite khmer). Then we calculate abundance distribution of the k-mers in the quality filtered reads using the pre-made k-mer counting table using `abundance-dist.py`. We followed the same approach to examine k-mer presence in assemblies.

Gene annotations using Prokka

We used prokka [**?**] to annotate the reference genome using `-outdir mprokka -prefix testasm -metagenome`. Then we parsed the testasm.tbl output file to get the coordinates of CDS genes. We then searched the alignments for how many genes were contained in those alignments.

Analyzing Assembly: Ambiguous, Best-Hit, and No misassemblies Approaches

We processed the alignments in three different ways: ambiguous, best-hit, and no-misassemblies.
In the ambiguous approach, we took into account all the alignments of a contig to the reference, even if alignments overlap in the reference or they are aligned to multiple locations in the reference.
In the best-hit approach, among all alignments of a contig, we took into consideration only the alignment with the best score.
In the no-misassemblies approach, we only counted contigs that have precisely one alignment to the reference.
In all approaches, we flag a base in the reference genome as "covered" if it is contained in a kept alignment. We define the duplication ratio as the percentages of bases in the reference covered by two or more kept alignments. We define misassemblies as those contigs that are divided into different parts when mapped to the reference. The number of misassembled contigs is equal to the number of aligned contigs (both totally and partially) in the ambiguous approach, minus the number of aligned contigs in the no-misassemblies approach.
All approaches have a non-zero duplication ratio within the reference because we do not explicitly discard contigs that map to the same location in the reference.

Results

The raw data is high quality

We trimmed sequences as described in Methods. We retained 7.1 Gbp in 108,422,358 paired-end sequences, and 36 Mbp in 520,403 orphaned reads. This quality trimmed ("QC") data set was used as the basis for all further analyses.

98% or more of the reference is present in the read data set

We next evaluated the fraction of the reference genome covered by at least one read (see Methods for details). Quality filtered reads cover 203,058,414.0 (98.76%) bases of the reference metagenome (205,603,715 bp total size). Figure 1 shows the cumulative coverage profile of the reference metagenome, and the percentage of bases with that coverage. Most of the reference metagenome was covered at least minimally; only 3.33% of the reference metagenome had mapping coverage <5, and 1.24% of the bases in the reference were not covered by any reads in the QC data set.
In order to evaluate reconstructability with De Bruijn graph assemblers, we next examined k-mer presence for a k-mer size of 20. Of the 174m 20-mers in the reference data set, 98.7% were present in the data set and 95.5% of them occurred with abundance 5 or greater in the quality filtered read data set.

MEGAHIT is the fastest and lowest-memory assembler evaluated

We ran three commonly used metagenome assemblers on the QC data set: IDBA-UD, SPAdes, and Megahit. We recorded the time and memory usage of each (Table 1). MEGAHIT outperformed both SPAdes and IDBA-UD considerably, producing an assembly in one hour – approximately 17 times faster than IDBA and 42 times faster than SPAdes. MEGAHIT used only 34.4 GB of RAM – 1/5 to 1/11th the memory used by IDBA and SPAdes, respectively.

Much of the reference is covered by the assemblies

We next evaluated the extent to which the assembled contigs recovered the "known/true" metagenome sequence by aligning each assembly to the reference (Table 2). All three assemblers generate contigs that cover more than 89% of the reference metagenome at high identity (99%) with little duplication (0.55 -1.0%) (see "Ambiguous approach" in Table 2). If we relax the identity threshold to 95%, then the assembled contigs cover more than 94% of the reference metagenome.

When we use only the highest-scoring alignment at 99% identity, we find that the reference coverage drops to 57-68%, depending on the assembler ("Best hit approach", Table 2). The reference coverage from highest-scoring alignments does not substantially increase when the identity threshold is relaxed to 95%.

At 99% identity with the ambiguous approach, approximately 6.23% of the reference is covered by no contig from any of the three assemblies; we discuss this in more detail below.

The generated contigs are broadly accurate

When counting only the best alignment per contig at a 99% identity threshold, more than 72% of contigs align to the reference completely, i.e. across the whole length of the contig (Table 3, "Best hit", Totally Aligned Contigs %). If we allow multiple alignments per contig, then more than 80% of contigs align completely (although not contiguously) to the reference (Table 3, "Ambiguous", Totally Aligned Contigs %). Approximately 4-9% of the contigs align only partially, and the remaining 8 - 10% of contigs do not align at all to the reference (discussed in detail below).

The assemblies contain most of the raw data

The assemblies also represent the majority of the reads (99.7%) and the majority of the abundance-5 20-mers (93.7% of the 198m high-abundance 20-mers in the reads), suggesting that the assemblies represent the underlying content of the reads very well.

Most genes within the reference metagenome are contained within contigs

The reference genome has 188,880 CDS with 91,806 annotated as genes, based on a Prokka annotation (see Methods). Using the "ambiguous" alignment approach and 99% identity, the IDBA-UD assembly contained 82,791 (95.20%) of the reference genes, while SPAdes contained 83,475 (94.44%), and MEGAHIT contained 80,256 (94.59%) of the reference gene coordinates.

The portions of the reference metagenome that are not reconstructed are not present in the read data set

We identified XX bases in the reference that had no match (at 99% identity with the ambiguous mapping approach) in any of the assemblies, and evaluated their base coverage. XX (19.7%) had no coverage in the reads, and a total of XX (29.2%) had coverage less than 5.

Large portions of several reference genomes are not assembled by any assembler

A number of the genomes in the reference metagenome had many missing bases in the assemblies (Table 4). In three extreme cases, Shewanella baltica_OS223, Fusobacterium nucleatum, and Desulvovibrio contribute, 18.25%, 16.95%, and 12.85% of the common uncovered bases respectively.

Many assembled contigs do not align to the reference metagenome

Depending on assembler, between 8.04% and 10.87% of the assembled contigs do not align anywhere in the reference metagenome.

6.49% of reads mapped to the unaligned contigs of IDBA. Only 33.01% of those reads mapped to the reference. (2.14% of all the reads). 6.23% of reads mapped to the unaligned contigs of SPAdes. Only 28.97% of those reads mapped to the reference. (1.80% of all the reads). 5.87% of reads mapped to the unaligned contigs of Megahit. Only 24.80% of those reads mapped to the reference. (1.45% of all the reads) For each assembly, approximately 5m quality-filtered reads map only to the unaligned contigs (and nowhere in the reference).

For IDBA QC, the reads that aligned to the unaligned contigs but not to the reference has coverage bases 27.21% <5 in the unaligned contigs. 75.72% has coverage >0.

For SPAdes QC, the reads that aligned to the unaligned contigs but not to the reference has coverage bases (3,109,539) 24.04% <5 in the unaligned contigs. 79.10% has coverage >0.

For Megahit QC, the reads that aligned to the unaligned contigs but not to the reference has coverage bases (2,578,896) 20.18% <5 in the unaligned contigs. 83.08% has coverage >0.

All three assemblers recover most of the reference

XX% of the reference metagenome is recovered by all three assemblers ("common covered") with relatively little duplication. (99% identity, ambiguous alignments allowed).

All three assemblers fail to recover 6.23% of the reference

(2,518,234) of the common uncovered bases has zero coverage (19.66%), while (3,739,532) of the common uncovered bases has coverage <5 (29.2%).

At the margins, the three assemblers differ at about 1% in recovery

IDBA, SPAdes, and MEGAHIT each uniquely recover about 1% (0.64 -1.92%) of the reference metagenome, and each

uniquely fail to recover about 1% (1.22-1.79%) of the reference.

Pending text: To add or not

Figure 3 shows the number of contigs using different minimum contigs cutoff. The figure shows that Megahit [**?**] has more fragmented contigs in the assembly. The small contigs size leads to best alignment; Megahit has the highest number of uniquely covered bases, highest genome coverage, and lowest unalignment.

Discussion

Assembly recovers basic content well

The majority of the reference metagenome is recovered by all three assemblers: 89% or more of the reference metagenome is contained within each assembler's output at 99% identity, and 94% can be recovered if we relax the identity threshold to 95%. This is close to the measured maximum reconstructability based on read mapping and k-mer presence: 98.76% of the reference is covered by at least one read, and 98.7% of the genome is present in k-mers of size 20. and 98.7 are from SRR606249.qc.coverage and SRR606249.qc.dist respectively

The contigs generated align well to the reference metagenome: 80% (or more) of contigs align to the reference metagenome across the whole length of the contig, and another 4% (or more) of the contigs are entirely contained within the reference metagenome, although they do not align to only one location in the reference. This could be caused by misassembly (a computational error) or by rearrangements in the source DNA (in which case the reference is incorrect).

More than 6.23% of the reference metagenome is missing from all of the assemblies when 99% similarity is required, and large portions of the missing sequence are from a few genomes – in some cases, close to a third of the source genome is missing from the assemblies.

of the unassembled regions of the source genome is low to nonexistent , while other portions of the source genomes are well represented within the reads; these regions were simply not sequenced, either because they were missing from the input DNA or because they challenge the sequencer. we have only 29.2

Assembly produces content not in the reference metagenome

In addition to recovering most of the content of the reference metagenome, all three assemblers generate many contigs (approximately 10% of the total) that do not map to anything in the reference. While some could be misassemblies, many of the reads that map to these contigs do not map anywhere in the reference, while many of the contigs have coverage >XX, suggesting that they are present within the source DNA at an abundance that is similar to many of the genomes in the mock community.

Whatever their true identity, these contigs are not present in the reference metagenome. Because this is a mock metagenome for which isolates were grown and individually extracted prior to being combined for sequencing, we believe that these extra genomic contigs must come from one or more contaminants. We cannot rule out contamination from kits, but because large amounts of DNA were used to create the mock community, it seems unlikely that these are due to trace contaminants created by PCR (as can happen in amplicon studies); they are most likely from contaminants grown together with the isolates.

Different assemblers differ, but not by much

The three assemblers differ very little in recovery of basic content, as judged by reference metagenome alignments: at 99% identity, XX% is recovered in common, with 0.64 to 1.92% specific to each assembler (and a total of XX% being recovered by one or two, but not all three, of the assemblers). None of the assemblers include all of the sequence produced by the others.

In practice, then, we do not see a significant advantage to one assembler over another in terms of recovering the known reference, and we speculate that (for this data set), there is little advantage to combining assemblies from multiple assemblers, since very little extra content will be recovered.

Conclusions

Assembly works well. There is no big difference between assemblers' performance in terms of assembly quality. In terms of cost, Megahit is much faster and utilizes less memory.

Author contributions

In order to give appropriate credit to each author of an article, the individual contributions of each author to the manuscript should be detailed in this section. We recommend using author initials and then stating briefly how they contributed.

Competing interests

No competing interest to our knowledge.

Grant information

This work is funded by Moore and NIH.

Acknowledgments

Michael R. Crusoe